New Constructive Approach to Covert Channel Modeling and Channel Capacity Estimation

Zhenghong Wang and Ruby B. Lee

Department of Electrical Engineering, Princeton University
Princeton, NJ 08544, USA
{zhenghon, rblee}@princeton.edu

Abstract. In this paper, we examine general mechanisms that a covert channel may exploit and derive new minimum requirements for setting up a covert channel. We also propose a new classification of covert channels based on our analysis. Unlike the non-interference approaches, our approach is constructive, allowing the direct examination of system architectures at different abstraction levels for the presence or absence of the mechanisms that can be exploited to create covert channels. Also, unlike past research on covert channel capacity estimation which employed a synchronous channel model, we point out that covert channels are generally non-synchronous. To capture the asynchronous nature of covert channels, we propose the deletion-insertion channel model as a more general basis for covert channel capacity estimation. This enables modeling the effects of system behavior on covert channel capacity, leading to a more accurate upper bound of the resulting channel capacity.

1 Introduction

A covert channel is often referred to as a communications channel that is neither designed nor intended to transfer information [2]. It allows information flows that violate security policies, utilizing only legitimate operations and shared resources of a system - in unintended ways. Covert channels have been acknowledged as serious threats [3].

Research in covert channels covers several subfields, including covert channel identification, channel capacity estimation, covert channel handling and mitigation. Covert channel identification finds illegal information flows, i.e., information flows that violate security policies. Though simple in concept, it is hard to perform in practice. While some covert channels may be easy to find, it is extremely difficult to find all. After a covert channel is identified, its channel capacity [8] should be analyzed. This has been done either using information theory to calculate the channel capacity mathematically, or using experimental means [4].

In this paper, we focus on the identification of covert channels and channel capacity estimation. We first propose a new covert channel model which allows analysis of all types of covert channels on a given abstract system specification. This includes all known covert channels, including the traditional "storage" and "timing" channels as

J. Zhou et al. (Eds.): ISC 2005, LNCS 3650, pp. 498-505, 2005.

well as new types of covert channels. We then present a new definition of the minimal requirements for setting up a covert channel and a new classification of covert channels. In estimating covert channel capacity, we point out that covert channels are asynchronous in general. By examining general synchronization mechanisms, we show the impact of legitimate information flows on covert channel capacity.

This paper does not consider covert information transfer techniques such as embedding information into image files or network packet headers, also called steganographic, “information hiding” techniques. Also, due to space limitations, we have had to omit several details, which are available in our full paper [16].

2 Related Work

The notion of covert communication was first introduced in [2]. In 1983, Kemmerer proposed one of the most widely used methods in covert channel identification [1]: the shared resources and the operations that are used to view and modify resources are first enumerated, a *Shared Resource Matrix (SRM)* is then constructed and each resource is carefully examined to determine whether it can be used to transfer information covertly. The *non-interference* approach was introduced by Goguen and Meseguer [6] and has been applied to a number of systems including the Honeywell Secure Ada Target (SAT) abstract model [7]. A view of the system state must be constructed for each user. One user process is non-interfering with another when the output observed by the second user process is unchanged if all inputs from the first user process, from the initial state, are eliminated as though they had never occurred.

Our work is motivated by the fact that despite significant past research on covert channels, it is still not clear how covert channels can be set up in general, leading to categorization of covert channels and related parameters that may be ambiguous [13][14]. Our study indicated that the term “time”, commonly used in analyzing covert channels, is a source of ambiguity since it can not be rigorously defined. We therefore propose to model covert channels without depending on ambiguous definitions of time. Secondly, the non-interference analysis normally models a system as an “interface”, which only specifies the requirements on a system without giving hints on how to implement the system. In our work, we reveal the general mechanisms by which one subject can interfere with the other. Our constructive approach is complementary to the interface model, and has direct implications for system architecture design. Thirdly, the success of an interface model relies on the correct definition of the interface. However, it is inadequate to prove the security with respect to a high-level abstract interface only. The interface has to be defined with considerations ranging from the highest-level abstract specifications all the way down to the lowest-level hardware implementations. This is complicated and error prone. It makes the design less portable and is not suitable for hierarchical development. In our work, we analyze the mechanisms in a general way so that they can be applied at each abstraction level.

In measuring the significance of a covert channel, Millen first established a connection between Shannon’s theory of communications and information flow models [8]. In 1989, he modeled an important class of covert channels as finite state machines [12]. Moskowitz [9] studied a class of covert channels that is discrete, noiseless and memoryless, called the Simple Timing Channels (STC) in 1994. In 1996, he analyzed a class of covert timing channel, called the timed Z-channel, and showed the bound on its capacity [10]. Comprehensive information and examples about covert channel analysis can be found in Virgil Gligor's Covert Channel Analysis guideline [4] and McHugh's Covert Channel Analysis chapter [5].

Past research on covert channel capacity estimation typically assumed that the covert channels are synchronized. We point out that covert channels are typically asynchronous, and propose the deletion-insertion channel model as a general basis of covert channel capacity estimation. This approach can provide more accurate capacity estimation since it takes asynchronous effects into account. It also enables the evaluation of the effects of different system designs on covert channel capacities.

3 Proposed Model

We first define system abstraction levels for covert channel analysis, then derive minimum requirements for setting up a covert channel. A new classification of covert channels is also proposed.

3.1 System Abstraction Level

To deliver information to the receiver, the sender must be able to do something that the receiver can "see". However, simultaneously considering all such mechanisms at all levels of the system is intractable. Rather, we propose analysis of one abstraction level at a time, using general mechanisms that can be adapted to any level.

By "see", we mean any methods by which the observer can learn the status, or value, of an object. This definition is not rigorous however. For example, when considering what can be seen by a program running on a computer system, we may agree that zeros and ones in the registers and memory are visible. But how about the voltage at the register's port and the charge in the capacitor of a DRAM cell? One may argue that they can be "seen" because they are the physical representations of those zeros and ones. But others may disagree since the program should only work in a logical world. This argument indeed reveals a useful fact: the visibility of a variable to the observer depends on the abstraction level. We therefore define:

Definition 1: By *see* we mean any methods provided at the current abstraction level by which the observer can learn the value of an object.

Definition 2: The *visible space V* of an observer is the set of all objects that the observer can *see*.

No matter what physical mechanisms the sender uses to deliver information, eventually the invoked changes will appear in the visible space of the receiver at the current abstraction level. Mechanisms that can not invoke changes in the visible space will not be able to transmit information at the current abstraction level, though they may be utilized at other levels. We therefore can restrict our analysis to one abstraction level at a time, without worrying about other levels.

In this paper we model a computer system as a state machine which contains active subjects, e.g., running programs, and passive objects, e.g., the data that the programs are working on. The passive objects form the machine's state and the subjects update this state. Given an abstract system specification, we can derive objects and atomic operations and define a subject as follows:

Definition 3: A subject is a sequence of "atomic" operations which take some objects as input and update some objects as output.

Definition 4: An operation is "atomic" if the state-updating process of the operation is indivisible.

All running programs in a system are modeled as subjects. A piece of hardware that generates data can also be modeled as a subject working on a specific object, at an appropriate abstraction level. The sender and the receiver are subjects, each of which may include multiple subjects. In addition, we define a *stranger* as follows:

Definition 5: A stranger is a third party that is also able to make changes in the *visible space* of the receiver.

The sender has no control over a stranger. A stranger may be totally unaware of the communication between the sender and the receiver. We introduce the notion of a stranger because it plays an important role in setting up a covert channel when the sender does not have "write" access to the receiver's visible space.

3.2 Minimum Requirements for Setting Up a Covert Channel

Theorem 1: If the sender is able to invoke change(s) in the *visible space* of the receiver, a covert channel may exist.

Proof: Consider the receiver and its *visible space* as a state machine. If the existence of the sender can change the execution trace of the receiver, we say that the receiver can learn information from the sender, i.e., a covert channel may exist. In this case**,** since the sender is able to invoke change(s) in the *visible space* of the receiver which is the state of the state machine, the future trace of the receiver can be changed, i.e., a covert channel may exist. □

Theorem 2: If the sender is able to change when an object is updated relative to the observation made by the receiver, a covert channel may exist.

Proof: Let OP_n denote the nth *operation* of the receiver. OP_n takes object OBJ_i as its input and updates OBJ_o as output. Let $OBJ_i(k)$ denote the kth update on OBJ_i. If the sender is able to control the update time of OBJ_i so that the update may occur either before or after the execution of OP_n, i.e., it can feed OP_n with either $OBJ_i(k)$ or $OBJ_i(k\text{-}1)$ as the input, the output of OP_n can be changed under the control of the sender. Therefore a covert channel may exist. □

These two theorems can be regarded as the minimum requirements for setting up a covert channel. Further discussions, including how they differ from Kemmerer's minimal requirements [1], and proof of Theorem 3 below are given in our full paper [16].

Theorem 3: A necessary and sufficient condition for setting up a covert channel is that the sender has either one or both of the abilities described in Theorems 1 and 2.

3.3 General Mechanisms and Covert Channel Classification

Table 1 summarizes our proposed classification of covert channels, based on the general mechanisms in Theorems 1 and 2. The first mechanism (Theorem 1) involves changes in the visible space of the receiver, which can be regarded as spatial information, resulting in what we call *spatial channels* (first two rows in Table 1). The second mechanism (Theorem 2) involves the change of the order of events which can be regarded as temporal information, resulting in *temporal channels* (last two rows in Table 1). While these seem similar to previous classifications of storage and timing channels, our contributions are to base them on unambiguous definitions at each system abstraction level (section 3.1), and to refine them based on further subdivision into *value-based* and *transition-based* spatial and temporal covert channels.[1] This provides not only clarification for some types of storage channels, but also reveals a new class of timing channels not previously identified.

Table 1. New Classification of Covert Channels

Class	Setup Mechanism
Value-based spatial channel	The sender is able to change the value(s) of one or more objects to the value(s) it wants. The receiver extracts information based on the value(s) it sees.
Transition-based spatial channel	The sender can determine whether or not modifications on one or more objects will be invoked. The receiver learns information from whether a change occurs or not.
Value-based temporal channel	The sender is able to learn or predict the value of an object and have control on when the receiver makes observations of that object. The sender keeps waiting until a proper value appears on the object. The sender then tries to let the receiver make an observation. Information is extracted based on the observed values.
Transition-based temporal channel	The sender can control the order of modifications on one or more objects, *relative* to observations made by the receiver. The receiver extracts information from the order of such events instead of the values of objects.

Our *value-based spatial channels* are typical covert storage channels, and hence not new. However, our *transition-based spatial channel* clarifies the fact that a covert storage channel can be created *indirectly* without needing the sender to have any control on the value of the object. For example, the sender need not have write access to the object that the receiver sees. This was not always clear in previous work.

Our *transition-based temporal channels* are like the timing channels disscussed in Wray's dual-clock analysis [13]. The data flowing through the channel are purely determined by the relative order of multiple clocks. However, our *value-based temporal channels* are a new class of channels. To our knowledge, this class of covert channels has not been identified in past work.

A simple example of a value-based temporal channel follows: Assume that subjects S and R are two applications in a mobile computing device used as a security

[1] Although a transition can be modeled as the difference of old and new values, this explicit classification is helpful in analyzing real covert channels.

token. S is not allowed to communicate with R but has certain control on when R is activated. The token records the usage information of the card, e.g., number of uses or frequency of usage, which is public to all subjects via either a software or hardware mechanism independent of S and R. S can then try to activate R whenever it sees a value that it wants to send to R, i.e., S can select a sequence of values for R to see.

Unlike non-interference approaches, our constructive approach has direct implications on system design. For example, Theorem 2 implies that if a system allows the operations of a subject to complete in a non-unique order (e.g., out-of-order disk access optimizations), or there are strangers in the system (e.g., the token usage recording mechanism above), covert channels may exist. Also, the setup mechanisms we propose can facilitate the investigation of real exploit scenarios [16].

4 Covert Channel Capacity Estimation

Unlike communication systems where synchronization is often specifically designed for reliable communication, synchronization mechanisms are usually not available for covert channels. Also, the communicating parties in covert channels often have limited or even no control in choosing the proper time to perform an operation, e.g., send a symbol to the channel or sample the channel to receive a symbol. Therefore a symbol sent by the sender may be dropped and the receiver may receive symbols that the sender never sent. Such a channel can be modeled as a deletion-insertion channel [11].

Theoretical research has shown that a channel with symbol insertions and dropouts is hard to use and inefficient. Past work on deletion-insertion channels showed that although such channels have non-zero capacity, in practice they are hard to use. However, this does not mean that the capacity of a covert channel is always low. As a deletion-insertion channel is a channel with memory, adding feedback to such a channel can increase its channel capacity. Hence, the impact of other information flows on channel capacity should also be considered since such information flows are often legitimate flows in the system and therefore can always be exploited.

4.1 Construction of Synchronization Mechanisms and Capacity Estimations

Figure 1 shows two ways to achieve synchronization utilizing extra resources in addition to the asynchronous covert channel: using feedback or using common events. To estimate the capacity of the channel with feedback, we first give two definitions:

Definition 6: A *binary deletion-insertion channel* is a channel with four parameters: P_d, $P_{i,}$ P_t and P_s, which denote the rates of deletions, insertions, transmissions and substitutions, respectively.

Definition 7: An *extended erasure channel* is a channel where symbols may be inserted and/or dropped but the locations of all insertions and dropouts are known.

As shown in [15], since an extended erasure channel knows more information than a deletion-insertion channel, the capacity of an erasure channel with feedback will be higher than or equal to the capacity of a deletion-insertion channel with feedback. However, since an erasure channel is a memoryless channel, adding feedback to it will not increase its capacity. Therefore the capacity of the erasure channel is an upper bound of the capacity of the deletion-insertion channel with feedback. Furthermore, since such an upper bound can be practically achieved using simple protocols [15], it is indeed the capacity of the deletion-insertion channel with feedback.

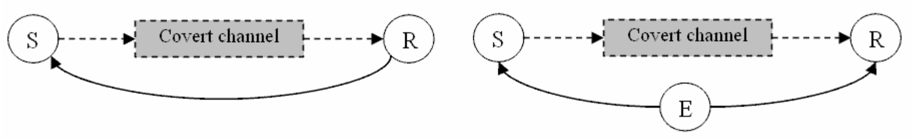

Fig. 1. Two general synchronization mechanisms

The capacity estimation of the deletion-insertion channel with common events is not trivial. However, it can be shown that its capacity is no greater than the capacity of the channel with feedback [15]. In summary, the capacity C of a covert channel is the capacity of the corresponding erasure channel, i.e.,

$$C = N(1\text{-}p_d) \quad (1)$$

where N is the number of bits encoded in each channel symbol. Since the deletion probability p_d is often determined by the system design, e.g., the scheduling algorithm, our approach provides a way to evaluate the impact of system design on covert channel capacity. Our work also reveals that other information flows may increase the capacity of a covert channel. This has interesting implications for a multi-level security (MLS) system. Since the legal information flow (from low to high) can serve as a perfect feedback path, one may always exploit it to achieve the channel capacity. In other words, covert channels in MLS systems can be relatively easy to exploit in general and can be quite fast [15][16].

5 Conclusions

We have proposed a new covert channel model which allows analysis of all types of covert channels at each system abstraction level. This includes all known covert channels, including the traditional “storage” and “timing” channels, as well as new types of covert channels. We present a new definition of the minimal requirements for setting up a covert channel and a new classification of covert channels. This exposes a new class of “value-based temporal” channels.

In estimating covert channel capacity, we point out that covert channels are generally asynchronous. We propose the deletion-insertion channel model as a more general basis of channel capacity estimation and consider the impact of other information flows. This approach can provide more accurate capacity estimation and, more importantly, can provide a means for evaluating the effects of different system designs, e.g., the scheduling algorithms, on covert channel capacities. It also shows interesting implications of legitimate information flows in certain systems such as MLS systems.

Acknowledgements

This work is supported in part by NSF CCR-0430487 and DARPA. We thank Prof. Tim Levin at NPS and the anonymous reviewers for their comments.

References

1. R.A. Kemmerer, “Shared Resource Matrix Methodology: An Approach to Identifying storage and Timing Channels,” *ACM Transactions on Computer Systems*, vol. 1, issue 3, pp. 256-277, August 1983
2. B.W. Lampson, “A Note on the Confinement Problem,” *Communications of the ACM*, vol. 16, issue 10, pp. 613-615, October 1973
3. J. Millen, “20 Years of Covert Channel Modeling and Analysis,” *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 113-114, May 1999
4. National Computer Security Center, “A Guide to Understanding Covert Channel Analysis of Trusted Systems,” NCSC-TG-30, November 1993, available at http://www.radium.ncsc.mil/tpep/library/rainbow/
5. John McHugh, “Covert Channel Analysis: A Chapter of the Handbook for the Computer Security Certification of Trusted Systems,” December 1995, available at http://chacs.nrl.navy.mil/publications/handbook/
6. J.A. Goguen and J. Meseguer, “Security Policies and Security Models,” *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 11-20, April 1982
7. J. T. Haigh, R.A. Kemmerer, J. McHugh, and W.D. Young, “An Experience of Using Two Covert Channel Analysis Techniques,” *IEEE Trans. on Software Engineering*, vol. 13, issue 2, pp. 157-168, February 1987
8. J.K. Millen, “Covert Channel Capacity,” *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp. 60-66, April 1987
9. I.S. Moskowitz and A.R. Miller, “Simple Timing Channels,” *Proceedings of IEEE Computer Symposium on Research in Security and Privacy*, pp. 56-64, May 1994
10. I.S. Moskowitz, S.J. Greenwald, and M.H. Kang, “An Analysis of the Timed-Z Channel,” *Proceedings of IEEE Computer Symposium on Security and Privacy*, pp. 2-11, May 1996
11. M.C. Davey and D.J.C. Mackey, “Reliable Communication over Channels with Insertions, Deletions, and Substitutions,” *IEEE Trans. on Information Theory*, vol. 47, no.2, pp. 687-698, February 2001
12. J.K. Millen, “Finite-State Noiseless Covert Channels,” *Proceedings of the Computer Security Foundations Workshop II*, pp. 81-86, June 1989
13. J. C. Wray, “An analysis of covert timing channels,” *Proceedings of IEEE Computer Symposium on Research in Security and Privacy*, pp.2-7, May 1991
14. Zhenghong Wang and Ruby Lee, “Separating data and signaling channels in modeling covert channels,” *Princeton University Department of Electrical Engineering Technical Report CE-L2004-003*, November 2004
15. Zhenghong Wang and Ruby B. Lee, “Capacity Estimation of Non-Synchronous Covert Channels,” *Proceedings of the 2nd International Workshop on Security in Distributed Computing Systems*, June 2005
16. Zhenghong Wang and Ruby B. Lee, “New Constructive Approach to Covert Channel Modeling and Channel Capacity Estimation (full paper),” *Princeton University Department of Electrical Engineering Technical Report CE-L2005-004*, April 2005