COVERT CHANNEL CAPACITY

Jonathan K. Millen

The MITRE Corporation
Bedford, MA 01730

Techniques for detecting covert channels are based on information flow models. This paper establishes a connection between Shannon's theory of communication and information flow models, such as the Goguen-Meseguer model, that view a reference monitor as a state-transition automaton. The channel associated with a machine and a compromise policy is defined, and the capacity of that channel is taken as a measure of covert channel information rate.

INTRODUCTION

One of the objectives of computer security is to prevent the unauthorized disclosure of information. Models of protection mechanisms and policies that espouse this objective fall into one of two categories: *access control* models or *information flow* models. Access control models are easier to relate to the design of computer systems, whether at a hardware, operating system, or application system level; but they have been perceived as inadequate for a real understanding of information protection, because of the existence of *covert channels* in systems that obey an apparently airtight access control policy. Covert channels exist because there are other sources of information in a computer system besides the storage objects to which access is controlled - for example, the very error messages that deny access to some objects.

Information flow models attempt to explain all possible ways in which information may be compromised in a computer system, with a finer-grained, microscopic view in which information is recognized when it occurs in an individual system register or program variable, or even at the bit level. Analysis techniques based on such models have been successful at finding covert channels, but they are still not perfect; in many cases they overestimate the actual information flows, flagging possible compromises that really don't exist.

This work was supported by the U. S. Government under contract F19628-86-C-0001.

Some models are aimed at defining information flow exactly, with necessary and sufficient conditions. These models share the philosophy that information flow in a computer security context is related to inference: if one user can, by observing outputs available to him, deduce something about inputs from another user, there has been some information flow. Conversely, if there is no information flow, the user's outputs would be the same regardless of the input from the other user. This approach was suggested in a paper by Jones and Lipton[1] investigating protection mechanisms in the context of computations, considered as functions rather than machines. This paper defined a security policy as an information-hiding function applied to the input, and said that a protection mechanism was "sound" if the computational values received by the user could have been obtained from such censored input.

One direction of development from the secure computation idea was to look at the computations occurring in high-level language programs, due to individual statements, subroutines, or the entire program. This led to Cohen's definition of strong dependency[2] between variables in a program, and to syntactically-based analysis techniques such as Denning's lattice model[3].

Other models regarded secure systems as automata or state-transition machines. One of the most influential of these is the Goguen-Meseguer model[4]; there have been others as well. These include the Feiertag model[5], of which the Goguen-Meseguer model is a generalization; the theory of constraints[6], which dealt with non-deterministic systems; the separability concept of Rushby[7]; and, more recently, the model in Sutherland[8].

The automata-based information flow models do not agree on their definitions of information flow. The answer does not appear to lie in a deeper modelling context, such as recursive-function models, but rather in an understanding of the assumptions and circumstances envisioned in each model. It makes a difference, for example, whether the computer system has been invaded by Trojan horses.

Another problem with the automata-based information flow models is that they do not measure the rate of information flow, but only whether it exists or not, regardless of how small it may be.

It has been suggested that the answers to some of these problems might be found in information theory, Shannon's probabilistic theory of communication over noisy channels. Certainly some of the terminology of information theory has been adopted, and authors of information flow models seem to believe that their models are the appropriate interpretation in the state-machine context of the concepts of information theory, although no one has established that connection explicitly. Information theory also holds out some hope of measuring the rate of information flow over covert channels.

This paper establishes a connection between Shannon's theory of communication and the state-machine models of information flow in computer systems. It turns out that it is not very hard to do so, given only the most elementary notions of information theory. The paper shows that the Goguen-Meseguer model, and others, stipulate the existence of information flow precisely when the corresponding probabilistic channel model would be shown to have non-zero capacity. It explains how assumptions about the environment, which distinguish different models, can be related to assumptions about channel characteristics. In particular, the Goguen-Meseguer model is contrasted with one proposed by Sutherland using a simple example. Finally, a method of measuring the rate of information flow is suggested, and illustrated on the same example.

Familiarity with basic concepts in probability theory, such as random variables and conditional probability, will be assumed. No knowledge of information theory is assumed; the relevant definitions will be given below. The context and motivation for these definitions may be found in any text in information theory, such as Gallager[9].

CHANNELS

In its simplest terms, information theory deals with a system, called a *channel*, having one input and one output, as shown in Figure 1. The input, X, and output, Y, are random variables, and, in our context, will be assumed discrete. A single *experiment* or *trial* consists in entering an outcome for X into the channel and observing the resulting outcome for Y. If the channel were perfect, the value of Y would be determined functionally by the value of X. Some encoding and/or decoding may be assumed to occur in the channel, so that Y does not have to take on the same values as X.

Usually the channel is noisy. This means that, given a value for X, the resulting value for Y is not determined, but, instead, has a probability distribution, which is determined by the characteristics of the channel.

How can we describe a computer system in these terms? We begin with a state-machine model of a multi-user computer system. It is a standard automaton model enriched with a set of users: X, Y, Z, who serve as a source and destination for inputs and outputs. A machine consists of a state set S, and sets of inputs I, outputs O, and users U, together with a transition function

$$\text{do}: S \times I \rightarrow S,$$

an input source function

$$\text{src}: I \rightarrow U,$$

and an output function

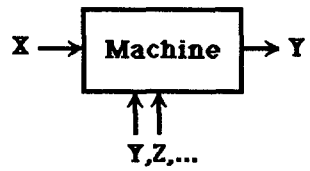
$$\text{out}: S \times U \rightarrow O.$$

The output function is a way for each user to test the state of the system. Different models have different conventions concerning when outputs are returned. The three main alternatives are (1) in the entry state of a transition caused by an input from the same user, (2) in the exit state of a transition caused by an input from the same user, or (3) in any state, when requested. An output request, in case (3), is treated as a kind of "input" that does not cause a state change.

Before going much further, we should indicate what the "system" is that is being modelled in this way. The machine in question is whatever system is being analyzed for information flow. In a computer security context, the machine would consist not only of the computer hardware, but also a security kernel layer, perhaps including also some trusted services. Inputs and outputs are defined where they pass through the security perimeter.

In a computer security context, we are interested in the information flow from a particular user, or group of users, to another user or group of users. For simplicity, let us not distinguish between different users in such groups, so that the issue is just information flow from one user to another. Figure 2 shows a situation in which we are looking at information flow from user X to user Y. It suggests that

we can think of the machine as a channel whose inputs come from X and whose outputs go to Y. Since the outputs observed by Y may depend also on the activities of users other than X, such as Y, Z, etc., those inputs are treated as contributing to the channel characteristics. Outputs to users other than Y exist, but are not shown because they are irrelevant to the flow from X to Y.



At first sight, it appears that one should identify inputs to the channel with inputs to the machine, but this would be wrong. The problem is that inputs to the machine, and outputs to users, occur in some ordering or time sequence, and that has a considerable effect on the observed outputs. We cannot assume that each input from X is followed immediately by an output to Y, nor do we know what the other users will be doing in the meantime. Hence the time between outputs to Y will be occupied by some sequence, perhaps empty, perhaps very long, of inputs from other users.

If we identify an output from the channel as a single output to user Y, it follows that the input from X to the channel should be the *sequence* of inputs to the machine that occurred since the last output to user Y. Thus, the random variable X, representing inputs to the channel from user X, ranges over I^*, the set of sequences of inputs to the machine. We should also remember that the "noise" in the channel will be due to the sequence of inputs from other users that occured during the same interval. The question of how these latter inputs are interleaved with the inputs from user X is also of concern; let us put this question off until later.

Besides X and Y, there are two other random variables associated with our model. Let V represent the sequence of all inputs from users other than user X during a trial. The combined sequence of inputs from all users during a trial is represented by a random variable W. Thus, the value of W is some interleaving of the values of X and V.

The random variables defined by the correspondence between the channel and the machine may be summarized as follows:

- A *trial* is a period ending with an output to Y.
- W is the sequence of all inputs during a trial.
- X is the subsequence of W consisting of inputs from user X.
- V is the subsequence of W consisting of inputs from users other than X, such as Y, Z, etc.
- $Y = \text{out}(t, \text{Y})$, the output to user Y in t, the state at the conclusion of the trial.

INFORMATION AND NON-INTERFERENCE

What we have described so far is a way of viewing a computer system as a communication channel, relative to information flow from a given user (or user group) to another given user (or disjoint user group). The information transmitted over the channel is defined in information theory as the *mutual information* $I(X;Y)$. When this quantity is maximized over all possible distributions for X, the result is the *channel capacity*. Mutual information and channel capacity are measured in bits; they can be related to information rate if the frequency of trials is known.

Knowing this much, we can already state the following theorem, and prove it after supplying the definition of non-interference:

Theorem: If X is non-interfering with Y, then $I(X;Y) = 0$, provided that X and V are independent.

The theorem states that non-interference is a sufficient condition for the absence of information flow. The additional independence assumption says that inputs from X are uncorrelated with inputs from other users; this assumption is necessary in order to prove the total absence of mutual information between X and Y.

We recall the definition of non-interference, in its simplest form[4]:

Definition: X is *non-interfering* with Y iff for all input sequences w in I^*,

$$\text{Y}(w) = \text{Y}(w/\text{X}),$$

where $\text{Y}(w) = out(t, \text{Y})$ if t is the state that results from applying the sequence of inputs in w, starting from the initial state; and w/X is the subsequence of w that remains when inputs from X are deleted ("purged").

It can be shown that the role of the "initial state" here is not essential; X is non-interfering with Y with respect to a fixed initial state, as defined here, if and only if X is non-interfering with Y with respect to all reachable states. Thus, in the context of a trial, we

may use the state of the machine at the beginning of the trial instead of the initial state.

Using the notation introduced in the above definition, we can say that, in general,

$$Y = \mathrm{Y}(W),$$

and if X is non-interfering with Y we have:

$$Y = \mathrm{Y}(V),$$

since

$$V = W/\mathrm{X}.$$

The reason for the assumption that X and V are independent is now clear: since Y is determined by V in this situation, any mutual information between X and V might turn into mutual information between X and Y. Inputs from X are not always uncorrelated with inputs from other users, but our main interest is in the machine itself; correlation between known and unknown inputs is uninteresting and obscures the contribution of any covert channel.

To prove the theorem that $I(X;Y) = 0$, we make use of a result from information theory to the effect that $I(X;Y) = 0$ if and only if X and Y are independent.

Let us proceed with the proof of the theorem. It suffices to show that

$$\Pr[Y = y \ \& \ X = x] = \Pr[Y = y] \Pr[X = x],$$

where $\Pr[Y = y]$ is the probability that the random variable Y has, as its observed value, the outcome y. But:

$$\Pr[Y = y \ \& \ X = x] = \Sigma_V \Pr[Y = y \ \& \ V = v \ \& \ X = x]$$

$$= \Sigma_{V:\mathrm{Y}(v)=y} \Pr[V = v \ \& \ X = x]$$

by non-interference;

$$= \Sigma_{V:\mathrm{Y}(v)=y} \Pr[V = v] \Pr[X = x]$$

by independence of X and V;

$$= (\Sigma_{V:\mathrm{Y}(v)=y} \Pr[V = v]) \Pr[X = x]$$

$$= (\Sigma_V \Pr[Y = y] \ \& \ \Pr[V = v]) \Pr[X = x]$$

$$= \Pr[Y = y] \Pr[X = x].$$

At this point it is reasonable to ask whether non-interference is also a necessary condition for the absence of information flow. It is not, and this was brought out by Sutherland[8]. We will use a simpler version of his example to contrast non-interference with other inference-based approaches, and then to illustrate the measurement of information flow.

A SIMPLE EXAMPLE

Consider a machine with two states 0 (initially) and 1. Suppose that there are three users, X, Y, and Z, and one input from each user, of the form (u, flip) where

$$\mathrm{src}(u, \mathrm{flip}) = u$$

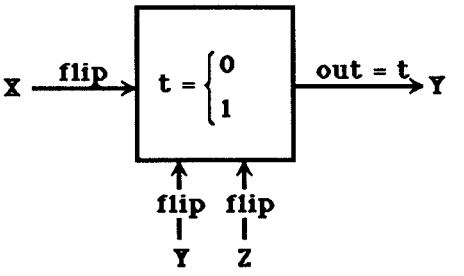
and flip is a constant command. Regardless of the user, each input flips the state:

$$\mathrm{do}(t,i) = 1 - t.$$

The output from a state is the state itself, again regardless of the user:

$$\mathrm{out}(t,u) = t.$$

The system is pictured below.



In this system, the output to Y depends entirely on the total number of inputs:

$$Y = \mathrm{Y}(W) = \mathrm{length}(W) \bmod 2.$$

It will be convenient, as we continue discussion of this example, to use the abbreviation:

$$\underline{w} = \mathrm{length}(w) \bmod 2$$

for any sequence w. Thus,

$$Y = \underline{W}.$$

It is easily determined that X is not non-interfering with Y. The input sequence W = (X, flip) serves as a counterexample, since:

$$Y(X, flip) = 1,$$

while

$$Y((X, flip)/X) = Y(nil) = 0,$$

where nil is the empty sequence.

We state this as an observation relative to this example:

Observation: X is not non-interfering with Y.

We now ask whether $I(X;Y) = 0$, under the assumption that X and V are independent. The answer is that it depends on the distribution of V. For, note that

$$\underline{W} = \underline{X} \oplus \underline{V}$$

where the sum is taken modulo 2. Thus, if $\underline{V} = 0$, the result will be that

$$Y = \underline{X},$$

in which case X and Y are clearly not independent. Note that, since V includes inputs from both Y and Z, user Y can control the parity of the length of V even without control over Z's inputs; knowledge of them, together with control over his own inputs, is sufficient. We record this as:

Observation: If $\underline{V} = 0$ then $I(X;Y) \neq 0$.

On the other hand, if user Y has neither control nor knowledge regarding user Z, it is quite possible that

$$Pr[\underline{V} = 0] = Pr[\underline{V} = 1] = 0.5.$$

We will describe this situation by saying that the parity of V is evenly distributed.

In this case X and Y are independent. We check this by comparing $Pr[Y = y]$ with $Pr[Y = y \mid X = x]$. First,

$$Pr[Y = y] = Pr[\underline{V} = y \ \& \ \underline{X} = 0] + Pr[\underline{V} = 1 - y \ \& \ \underline{X} = 1].$$

But X and V are assumed independent, so:

$$Pr[Y = y] = Pr[\underline{V} = y] \, Pr[\underline{X} = 0] + Pr[\underline{V} = 1 - y] \, Pr[\underline{X} = 1].$$

Substituting $Pr[\underline{V} = y] = 0.5$, we find that:

$$Pr[Y = y] = 0.5.$$

On the other hand,

$$Pr[Y = y \mid X = x] = Pr[\underline{V} = \underline{x} \oplus y] = 0.5.$$

Thus, Y is independent of X, and $I(X;Y) = 0$. Thus we have:

Observation: If X is independent of V, and the parity of V is evenly distributed, then $I(X;Y) = 0$.

Note that if Y and Z supply inputs independently, the parity of V is evenly distributed if the parity of input sequences from Z is evenly distributed, regardless of the inputs from Y.

To summarize the observations from this example, we have shown that non-interference implies that the information flow is zero. But non-interference is *pessimistic*, in the sense that it can indicate the (possible) existence of information flow in cases where no information flow actually exists. It does so because it does not take into account the possibility of independent users. Instead, it implicitly makes the worst-case assumption that inputs from all other users may be either known to the penetrator or under the control of a penetrator.

Now that we have characterized non-interference as "pessimistic", using an example suggested by Sutherland's paper, one might ask whether Sutherland's model is optimistic or pessimistic.

Sutherland's paper defines information flow in terms of "view functions". A view function maps "possible worlds" into values accessible to some user. Information flow between view functions is absent only when observation of the value of one does not permit one to infer any restriction in the set of possible values of the other, for any possible world.

In state machine instantiations of this idea, a view function shows some subsequence of inputs and/or outputs over any finite period. A security compromise occurs when there is information flow from a "hidden" view function showing only inputs from one user to a view function available to another user who is not cleared to see those inputs. If we call the latter view function the "visible" one, it turns out that this definition can be either optimistic or pessimistic depending on what the visible view function shows.

In our example, an appropriate hidden view function would be X, the inputs from X during a trial. If the visible view function were Y, this model would say that there is no information flow from X to Y, since

knowing either value for Y, 0 or 1, does not restrict the value of X, or even its parity; different parities for V can deliver either value for Y. This instantiation of the model is therefore optimistic.

On the other hand, if the visible view function showed both V and Y, an observation of it would determine the parity of X, restricting the possible values of X, so this instantiation of the model would conclude that there was information flow and hence a security compromise, making this instantiation a pessimistic one.

MEASURING INFORMATION FLOW

The mutual information $I(X;Y)$ measures the average quantity of information, in bits, that passes across the channel from X to Y in one trial, demarcated by an output to Y. We have noted above that this quantity should be calculated under the assumption that X is independent of V, the totality of other inputs. The example has shown that the quantity of information transmitted - indeed, its very existence - depends on the distribution of V. This is not surprising, since V, along with the structure of the underlying machine, determines the characteristics of the channel.

In information theory, the channel capacity is defined as the maximum of the mutual information over the possible distributions of X. This is done to obtain a number that is determined solely by the channel. In a communication context, the purpose of encoding is to change the distribution of X, and bring it closer to an ideal that brings about the maximum information transfer. In a computer security context, we may imagine that a Trojan horse aims to have a similar effect.

The calculation of channel capacity, and the way it depends on V, will be illustrated here using the example from the previous section. We have already shown that a certain class of distributions for V result in a channel capacity of zero. We now look at a more general case.

Our calculation of $I(X;Y)$ will be based on the formula:

$$I(X;Y) = H(Y) - H(Y|X),$$

where $H(Y)$, the entropy of Y, is given by:

$$H(Y) = -\sum_{y} \Pr[Y = y] \log \Pr[Y = y]$$

(Logarithms are base 2). The conditional entropy of Y given X is:

$$H(Y|X) = -\sum_{x,y} \Pr[X = x \ \& \ Y = y] \log \Pr[Y = y \mid X = x].$$

In our example, the value of $I(X;Y)$ can be determined from $\Pr[\underline{X} = 0]$ and $\Pr[\underline{V} = 0]$, keeping in mind that X and V are assumed independent. The calculations are straightforward but tedious and will be omitted. The resulting formula is stated using the abbreviations:

$$p = \Pr[\underline{X} = 0],\ p' = 1 - p$$
$$q = \Pr[\underline{V} = 0],\ q' = 1 - q.$$

The value of $I(X;Y)$ is given by:

$$\begin{aligned} I(X;Y) = &-(pq + p'q') \log (pq + p'q') \\ &+ pq \log q + pq' \log q' \\ &- (pq' + p'q) \log (pq' + p'q) \\ &+ p'q' \log q' + p'q \log q. \end{aligned}$$

Maximizing the value of $I(X;Y)$ over the possible values of $p = \Pr[\underline{X} = 0]$, we find that the maximum value is at $p = 0.5$, so the channel capacity C is given by:

$$C = 1 + q' \log q' + q \log q = 1 - H(\underline{V}).$$

In particular, if $\underline{V}$ is constant, the channel capacity takes on its maximum value of one bit per trial. On the other hand, as we observed above, if we have $\Pr[\underline{V} = 0] = 0.5$, the entropy $H(\underline{V})$ becomes equal to 1 and the channel has a zero capacity.

The relationship between "bits per trial" and "bits per second" depends on the frequency with which y observes the state. If the observations are so frequent that X has usually not had time to enter another input, the random variable X will be skewed toward X = nil, and the actual information flow will be far less than the channel capacity on a per-trial basis, though its effect on the flow in bits per second is not clear. On the other hand, if Y samples very infrequently, the bits per trial may be high, but the bits per second will tend toward zero as the seconds-per-trial period increases.

CONCLUSIONS

Given the construction of a channel from a machine as described above, the information flow in "bits per trial" between users X and Y is defined as the capacity of the channel, calculated under the assumption that inputs from X are independent of those from other users. A trial is a period of time between outputs to Y. It has been shown that if X is non-interfering with Y, then the information flow is zero. However, the information flow can be zero even when X is not non-interfering with Y.

The information flow over a covert channel can be calculated using the standard formulas for channel capacity, but the calculation is tedious even for a small example. An important fact about the channel is that the behavior of other system users contributes to the channel characteristics, resulting in widely varying figures for the capacity, depending on the assumptions that are made about inputs from those users.

Covert channel rate estimation techniques applicable to real systems will not be easy to develop. The following is an attempt to extrapolate the general principles and observations noted above, to suggest some possible characteristics of the analysis of a large system.

First, it appears that much of a covert channel rate analysis can be performed on a top-level formal specification of a reference monitor; the actual computation times associated with system functioning are needed, but can be kept symbolic until the end.

One should remember that X and Y are not likely to be actual users, but must be interpreted in a subtler way. X is a source of reference monitor calls containing information of interest to a penetrator, and may be only a small subset of the inputs arising from any particular user's process. Outputs to Y are reference monitor responses believed to be "revealing"; other, routine outputs to the same user group that will be ignored for purposes of tapping a covert channel can be ignored for purposes of defining a trial. The input behavior of other users falls into two classes: that which is assumed to be under the control of the penetrator or a Trojan horse, and that which is probabilistic. As in performance modelling, independent probabilistic inputs from other users should be treated as a "load" on the system, and reasonable distributions for them might be proposed.

Acknowledgement

Thanks to Josh Guttman for suggesting that trials should end with an output to Y.

REFERENCES

[1] A. K. Jones and R. J. Lipton, "The enforcement of security policies for computation," *ACM Operating Systems Review*, Vol. 9, No. 5 (November, 1975) pp. 197-206.

[2] E. Cohen, "Information transmission in sequential programs," *Foundations of Secure Computation* (Ed. R. A. DeMillo, et al.), Academic Press, New York, 1978, pp. 297-336.

[3] D. E. Denning and P. J. Denning, "Certification of programs for secure information flow," *Commun. ACM*, Vol. 19, No. 5 (May 1976), pp. 504-513.

[4] J. A. Goguen and J. Meseguer, "Security policies and security models," *Proc. of the 1982 Symp. on Security and Privacy*, IEEE, April 1982, pp. 11-20.

[5] R. Feiertag, "A technique for proving specifications are multi-level secure," *SRI Report* CSL-109, 1980.

[6] J. K. Millen, "Constraints: Part II, constraints and multilevel security," *Foundations of Secure Computation* (Ed. R. A. DeMillo, et al.), Academic Press, New York, 1978, pp. 205-222.

[7] J. M. Rushby, "Proof of separability, a verification technique for a class of security kernels," *International Symposium on Programming* (Turin, April 1982), Springer-Verlag, Lecture Notes in Computer Science No. 137, pp. 352-367.

[8] D. Sutherland, "A model of information," *9th National Computer Security Conference* (15-18 September 1986), National Bureau of Standards/National Computer Security Center, pp. 175-183.

[9] R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley and Sons, Inc., New York, 1968.