An Insecurity Flow Model

Ira S. Moskowitz and Myong H. Kang
Center for High Assurance Computer Systems
Information Technology Division, Mail Code 5540
Naval Research Laboratory, Washington, DC 20375 USA

ABSTRACT

We examine a new way of looking at security violations, called *insecurity flow*. We express our new paradigm via a formal mathematical model that combines elements of graph theory and discrete probability. Prior work by Moore and Shannon on building reliable circuits out of unreliable components, and the physics community's interest in dynamical systems, especially percolation theory, motivates our work.

1. A New Paradigm

We wish to analyze how insecurity may spread throughout composed protection schemes. System (information) insecurity can be investigated in many different ways. Previous work on this area has been concerned with information flow; by looking at the capacity of a covert channel [Mi], or investigating which variants of noninterference hold upon system composition [Mc]. This previous work is limited to multilevel security. The concept of illicit information flow is certainly important and well documented. However, we wish to investigate the concept of *insecurity flow*, not information flow. Thus, we put forth a new paradigm dealing with the abstract notion of an "invader" penetrating through "security holes" of various protective security domains. We present a formal model of this new paradigm of insecurity flow by using graph and discrete probability theory. Prior work exists using graph theory and/or probability theory [D1], [D2], [O], [Le], [LB] to model system reliability. However, those probabilistic models analyze how the probability of "breaking in" varies with time. Our paradigm views a "break in" as a 0/1 proposition. The breach either happens or does not happen. It does not, like the reliability-based work, happen asymptotically in time. In our paradigm, there is no concept of time other than one penetration following another. The time spent for an insecurity to flow past some protective layer is not a factor once we set the probabilities. We are simply concerned with whether or not an insecurity can or cannot make it past the protection. Our model is not concerned with whether an insecurity can "back track" out. The security violation is simply that the protective layers have broken down and the insecurity has been able to flow in. In other words, something only has to become "infected" or "burnt" for the violation to take place.

1997 New Security Paradigms Workshop Langdale, Cumbria UK
0-89791-986--6/97/ 9

An insecurity flow is similar to fire spreading through an entire forest or a liquid spreading through a porous material. Those topics have been analyzed in the physics world via percolation theory [W],[La]. We have extrapolated this idea of percolation to security by looking at the abstract notion of an insecurity flowing from a source to a sink. The source is the point where the invader starts out and the sink is the repository of the information that we are attempting to protect. Once the insecurity has reached the sink, the game is over. There is no concept of partial flow; the insecurity either makes it through, or it does not. We are not interested in noninterference and all of its children, since we are not composing MLS systems. We are attempting to protect something. What we are composing are the protective layers, not any underlying MLS systems.

2. Introduction

2.1 Protection Domains

A motivation for our paradigm is to mathematically model the effects of using different architectures of various protection domains (e.g. firewalls) to protect against insecurity flow in today's open and distributed computing environment. In this environment, physical access control alone does not adequately protect organizations' computing resources. Now organizations must protect their resources within globally networked environments. Organizations can protect their computing assets (e.g., information and resources) by establishing protection domains [C] which contain

groups of related components. Protection domains can be established by separating the components both physically and logically.

Each protection domain may possess a security policy and may receive insecurity flow protection via security mechanisms such as firewalls, domain type enforcement mechanisms [B], discretionary access controls, mandatory access controls, etc. Some organizations may choose to employ a combination of security measures instead of just one security mechanism.

When many entry points for insecurity flow exist in security protection domains, it becomes more difficult to analyze the system's vulnerability as a whole. Some questions for a security designer are:

1. Which is the most vulnerable path for insecurity flow? What can make this vulnerable path more secure?
2. Is there any redundant or useless security protection measure?
3. Is one strong security measure better than two weaker security measures (assuming that a strong security mechanism costs more than two weaker mechanisms)?

As computer systems grow more complex, it becomes harder to analyze the vulnerability of insecurity flow of systems using ad hoc methods. To codify our new paradigm, we introduce a formal model to analyze the insecurity flow of distributed computer systems. We model the probability of "security violation" that exists despite security protection mechanisms. The work of Moore and Shannon on reliable electrical components [M1,M2] inspired our model. After the presentation of this paper, we came upon very similar ideas in [K]. In particular it is the discussion of electrical circuits and De Montmort's theorem in Appendix A of [K] (which was also in the original 1946 edition) that is similar. Instead of current flow, we look at insecurity flow. We also introduce techniques that simplify the analysis of security vulnerability by decomposing a complex network into many simpler networks.

2.2 Independence

In this paper, we assume that every protection domain behaves in an independent manner. One reason for this is that it makes our mathematics easier because independent probabilities multiply. We are introducing a new concept in security. Therefore, we are taking the first steps by using simplifying assumptions. We will study more complex interactions in future work. In fact, the mathematics in this paper can be easily equipped to handle dependencies between protection domains by using conditional probabilities, instead of the atomic probabilities that we do use.

Another reason for independence is that independence models good architectural decisions. Arguing from the viewpoint of strength we would not want to have the compromise of one safeguard influence the penetration of another safeguard. If the violation of one protection domain influences the probability of breaking through a different protection domain either (1) drop the independence assumption, or (2) consider changing the design of the protection domains so they are independent. For example, if a specific password allows an insecurity to flow into one protection domain, one obviously would not want to use the same password in another protection domain.

3. Protection Domains & Heuristic Modeling

Protection domains contain groups of related components. Protection domains can be established by both physical and logical separation and can be organized as hierarchies (levels). For example, a protection domain, PD1, may contain two protection subdomains, PD11 and PD12 (see figure 1). Components within the smallest protection domains (the double indexed protection domains) are assumed to trust each other and thus have no need for security services.

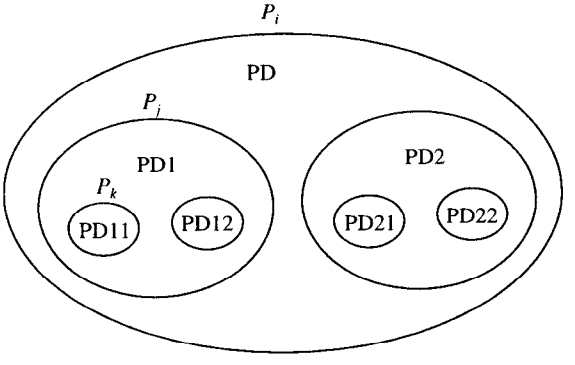


Figure 1: Hierarchies of protection domains

Let the probability of an insecurity flow through the security boundary of PD be P_i, the probability of an insecurity flow through the security boundary of PD1 be P_j, and the probability of an insecurity flow through the security boundary of PD11 be P_k. In this paper we view the probability assignments as givens. (Of course how one takes real systems and assigns probabilities is extremely important. For example, the difference if an intruder has five minutes or five months to attempt to break into a PD must be taken into account in assigning probabilities.) The strength of the system in figure 1,

with respect to insecurity flow, can be modeled as follows:

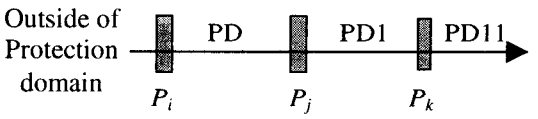


Figure 2: Serial connection

As mentioned, we assume that the organization has constructed protection domains to operate in an autonomous and distinct manner (hence, the probabilities of breaking in are independent). Thus, we see that an insecurity has probability P_i of its insecurity flowing into PD, P_iP_j of flowing into PD1, and $P_iP_jP_k$ of flowing into PD11. Note that $P_iP_jP_k$ is less than any of the P_α, $\alpha = i,j,k$. This accords with our intuitive notion that the concatenation of protection domains in a serial manner increases security. (Note, in [Le], the cascading effects of TCBs were analyzed.)

Let us consider another example. An organization may consist of several departments, with their computing resources connected by the organization's intranet. This heuristic modeling contains organizational assets, insiders, and outsiders. For example, the organization may set up its security measures as follows:

1. A firewall protects the organization's protection boundary, and
2. Simple access controls protect the departments' protection boundaries.

An organization may consist of geographically distributed and distinct protection domains, some of which may forge protected communication links into a virtual protection domain, as shown in figure 3. We assume that no intrusion is possible into communication links from the exterior; hence, the bold lines around them in figure 3.

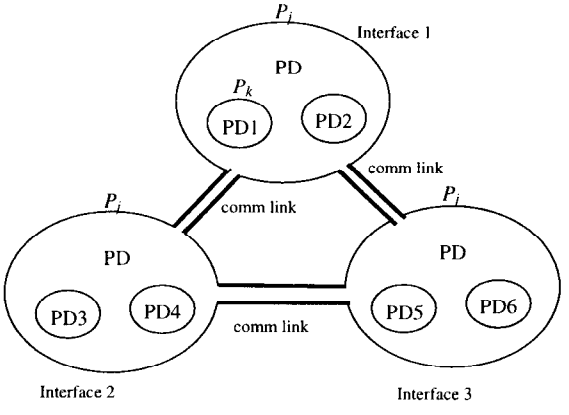


Figure 3: Virtual protection domain

Heuristically, we model the strength of protection domain PD of the system in figure 3 as follows:

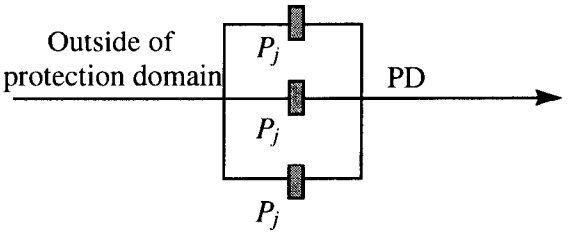


Figure 4: Three protection domains in parallel

Again, the protection domains are constructed in an independent manner. That is, the probability of an insecurity flowing through each interface is independent. However, for the sake of simplicity they all have the same probability P_j. In this *parallel* architecture, an insecurity has three separate routes (the interfaces) to flow into PD. Therefore, an intruder has a chance of flowing into PD in three different ways.

The event that there is a security violation (successful insecurity flow) is complementary to the event that there is not a security violation. The probability that there is not a security violation is $(1-P_j)^3$. Therefore, the probability that there is a security violation is $1-(1-P_j)^3$. Since $1-(1-P_j)^3 > P_j$, the above architecture provides *less* security than does a single protective domain.[1] Note that the same would hold if each interface had distinct probabilities of insecurity flow.

With respect to an insecurity flow of the system in figure 3, we model the strength of protection domain PD1 as follows:

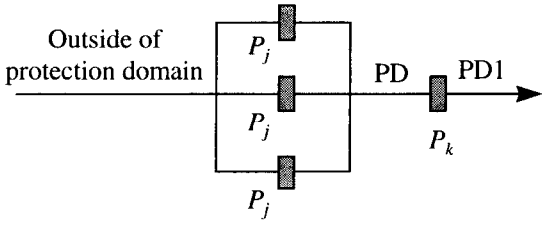


Figure 5: Combination of parallel and serial connection

How much protection does the architecture in figure 5 provide? This determination requires a more subtle probabilistic study of the various protection domains.

[1] Here, as in the rest of the paper, we ignore trivial comparisons when the probability values achieve the boundary values of zero or one.

Informal model:

We use an informal model to introduce the concept of circuit insecurity, a model that we develop formally in section 3. We view a security violation as something that flows, with links[2] that probabilistically allow an insecurity violation to flow from node to node. The nodes and links form a connected set that we refer to as a circuit. Heuristically, one can think of a circuit as a network of protection domains. The probability of a node being open is p; conversely, it is closed with probability 1-p. If the node is open, the insecurity flows through the node; if the node is closed, the violation does not pass through the node. (We do not consider the concept of partial flow.) The nodes behave independently with respect to their associated probabilities. The situation of dependent nodes is not addressed in this paper. Independence implies that a security violation does not "learn" as it passes through different protection domains. As previously noted, except for sequencing, time is not taken into consideration. This does not mean that time is not a consideration in assigning probabilities. The probability values are taken as assumptions. Of course when one does modeling, the actual time for an attack is what sets the probability values. More time for an attack would result in a higher probability value. We are concerned with the ordering of "events", not the actual time intervals between them. A special node exists, called the start node n_s, *from* which insecurity flows (the source), and a special node exists called the terminal node n_t, *to* which insecurity attempts to flow into. Every node n_i has a probability p_i of permitting the insecurity flow. The value of probability for n_s or n_t is always one. We define q_i as $1-p_i$. The term q_i is the probability that the node does not allow an insecurity flow. The higher the q_i value (conversely the lower the p_i value), the more secure our node is. For an insecurity to flow from n_s to n_t, at least one path must exist that connects n_s to n_t, such that each node successfully passes on the insecurity. Therefore, we see that the flow of an insecurity can be analyzed probabilistically.

Basic Examples, various circuits Σ:

SINGLE NODE:

$$n_s \longrightarrow n \longrightarrow n_t$$

For our preliminary discussions, we do not discuss the probabilities of n_s or n_t . (This is because they are implicitly assigned the probabilities of one, which are transparent to the calculations.) We concentrate our attention on node n with respect to probabilities. Node n has probability p of passing the insecurity to n_t. Node n has a given probability 1-p of being a successful security device. What if the level p is unacceptable? One might simply get a "better" node n, but the cost of a better node might well be prohibitive. Maybe two nodes identical to node n would give us the requisite security, e.g., a lower p value. Let us examine that scenario.

2 SERIAL:

$$n_s \longrightarrow n_1 \longrightarrow n_2 \longrightarrow n_t$$

For the insecurity to flow from n_s to n_t, both nodes n_1 and n_2 must advance the insecurity. For simplicity, in this example we assume that $p_1 = p_2 = .12$; in other words, we consider the nodes to be 88% secure. Therefore, the probability that the circuit Σ is insecure is

$$P(\Sigma \text{ insecure}) = P(n_1 \text{ passes insecurity}, n_2 \text{ passes insecurity}).$$

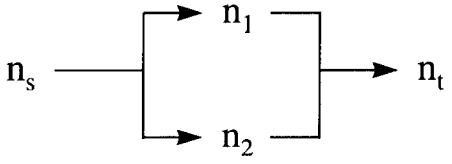
Since we are assuming independence between the nodes, this is simply

$$P(n_1 \text{ passes insecurity}) \bullet P(n_2 \text{ passes insecurity}) = (.12)^2 = .0144 < .12 .$$

Hence, the serial circuit is *more* secure than the single node circuit.

2 PARALLEL:

Now let us examine the case where there might be two choices of entrance into n_t.

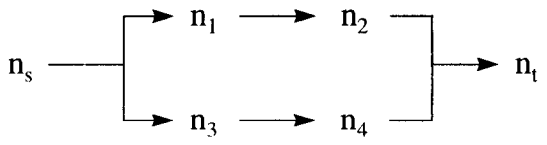


In this case, it is only necessary that the insecurity flow through either n_1 or n_2 (of course, it is possible that it flows through both nodes). Again, for simplicity's sake, we let $p_1 = p_2 = .12$.

$P(\Sigma \text{ insecure}) = 1\text{-}P(\Sigma \text{ secure})$
$P(\Sigma \text{ secure}) = P(n_1 \text{ secure}, n_2 \text{ secure}) = (1\text{-}.12)(1\text{-}.12) = .7744$
Therefore, $P(\Sigma \text{ insecure}) = 1 - .7744 = .2256 > .12$
Hence, the parallel circuit is *less* secure than the single node circuit. In other words, two parallel firewalls provide less protection than one firewall. Let us expand each node by two nodes in series.

[2] In section four, we will use edges instead of links.

2 PARALLEL/2 SERIAL



We assume that each node has $p_i = .12$.

P(Σ insecure) = 1 - P(Σ secure)
P(Σ secure) = P(top nodes secure, bottom nodes secure)
P(bottom nodes secure) = P(top nodes secure)
= 1 - P(n_1 insecure, n_2 insecure)
P(n_1 insecure, n_2 insecure) = P(n_1 insecure) P(n_2 insecure) =$(.12)^2$ = .0144, so
P(top nodes secure) = 1-.0144 = .9856
P(top nodes secure, bottom nodes secure) = $(.9856)^2$ = .9714
Therefore, P(Σ insecure) = 1-.9714 = .0286 < .2256
In fact, since .0286 < .12, we are even more secure than with one node! Of course, .0286 is greater than .0144, the insecurity value associated with two nodes in series.

Thus, if we have two nodes in parallel, we are always better off replacing them with the expanded 2 parallel/2 serial construction shown above.

Let us go through our basic architectures with a variable value of p for all of the node insecurities.

SINGLE NODE (1-circuit):
P(Σ insecure) = p

2 SERIAL (2S-circuit):
P(Σ insecure) = P(n_1 insecure, n_2 insecure) = p^2

2 PARALLEL (2P-circuit):
P(Σ insecure) = 1-P(Σ secure)
= 1-(P(n_1 secure, n_2 secure)) = $1-(1-p)^2$

2 PARALLEL/2 SERIAL (2P/2S-circuit):
P(Σ insecure) = 1-P(Σ secure)
= 1-(P(top branch secure, bottom branch secure))
= 1-(P(top branch secure) P(bottom branch secure))
= $1-(1-p^2)^2$

It is intuitive that a 2S-circuit should be more secure than a 1-circuit. Since $p^2 < p$, we see that this is true. A 2P-circuit should be less secure than a 1-circuit since we have two opportunities for insecurity flow. Since $1-(1-p)^2 = p(2-p) > p$, we see that a 2P-circuit is less secure than a 1-circuit. Using the p value of .12 from above, we showed that the 2P/2S-circuit is more secure than a 1-circuit. What guidance does our intuition give us? The serial part should make it more secure than the 1-circuit but the parallel part makes it less secure than the 1-circuit. The two factors must be studied together. Let us try some other values for p and see what happens.

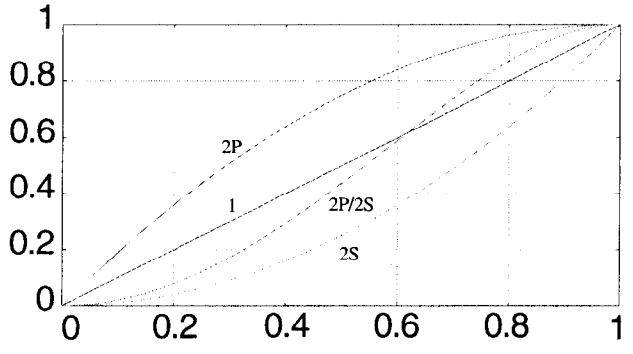
1	2S	2P	2P/2S
p	p^2	$1-(1-p)^2$	$1-(1-p^2)^2$
.12	.0114	.2256	.0286
.5	.25	.75	.4375
.88	.7744	.9856	.9491

For p =.12 or .5 the 2P/2S-circuit is more secure than the 1-circuit. However, for p = .88 the 2P/2S-circuit is less secure than the 1-circuit. One might assume that this is the case because .88>.5. However, .5 is not the magic number.

For p = .6 we may extend the above chart to

1	2S	2P	2P/2S
p	p^2	$1-(1-p)^2$	$1-(1-p^2)^2$
.12	.0114	.2256	.0286
.5	.25	.75	.4375
.88	.7744	.9856	.9491
.6	.35	.84	.5904

For p = .6, the 2P/2S-circuit is still more (just barely) secure than the 1-circuit. Therefore, a p value of .5 is not our holy grail. Of course, we just have to solve for $1-(1-p^2)^2 = p$ to obtain the transition value (approx.) .618 . We have taken this laborious path with the hope that the reader finds this phenomenon as surprising as the authors did when they first came upon it [M1, M2]. Therefore, before we use various node topologies to obtain the desired level of security, we must carefully consider what is going on.



Plot 1: Circuit Insecurity vs. node insecurity p

The comparison of the insecurity of 1, 2/S, 2P, and 2P/2S circuits is shown in plot 1. We see that 2P is always the least secure, 2S the most secure and 2P/2S lies in between the two. What is interesting is that, as a node becomes less secure ($p \rightarrow 1$) we see that 2P/2S switches from being more secure than a single node to less secure than a single node. The results are not constant across all p values. Therefore, any architectural solution that we use must take into account the probabilities of the nodes. We must also worry about situations where the p value changes from node to node. (Note that the 2P/nS circuit that we discuss later also has similar "alternating" behavior.)

Limiting Behavior:
If we have enough nodes linked together in series, can we achieve any desired degree of security?

$$n_s \longrightarrow n_1 \longrightarrow n_2 \longrightarrow \cdots \longrightarrow n_k \longrightarrow \cdots \longrightarrow n_t$$

If each node n_i, $i = 1,\ldots,k$ has the same probability p, $p < 1$, of being insecure then the probability that the circuit is insecure is $P(\Sigma \text{ insecure}) = p^k$. Since $\lim_{k \rightarrow \infty} p^k = 0$, we see that we can, by adding enough nodes into the serial link, get our security to within any tolerance of perfect security.

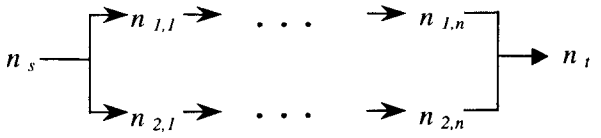
Since $P(\Sigma \text{ insecure}) = \lim_{k \rightarrow \infty} \prod_{i=1}^{k} p_i$ let us analyze $P(\Sigma \text{ insecure})$ when the p values of the nodes are not constant.

1. p_i bounded away from 1: There exists a number $B < 1$, such that $\forall i$, $p_i < B$, then $P(\Sigma \text{ insecure}) \leq \lim_{k \rightarrow \infty} B^k = 0$

2. 1 is an accumulation point for the $\{p_i\}$: An example of this is obtained by setting $p_i = 1\text{-}10^{-i}$, so $P(\Sigma \text{ insecure}) = \lim_{k \rightarrow \infty} \prod_{i=1}^{k} (1\text{-}10^{-i}) \approx .890$

This is not too startling because the p_i are quickly approaching one. However, it does tell us that stringing enough nodes together in a serial fashion will not always give us the security that we may desire. However, under the realistic assumption that the probabilities of the nodes are bounded away from one, we may string enough together to obtain the desired level of security. Of course, there is a finite limit to how many nodes (protection domains) can be strung together. Therefore, one must balance security against performance/cost/efficiency.

What are we to do if we are forced to have a parallel architecture in our circuit? Can we, by adding additional nodes into the circuit, reduce our insecurity? We have seen from above that turning a 2P into a 2P/2S circuit does increase our security, but it may not increase it sufficiently. Consider a 2P/nS circuit.



$P(\Sigma \text{ insecure}) = 1\text{-}P(\Sigma \text{ secure})$
$= 1\text{-}P(\text{top secure, bottom secure})$
$= 1\text{-}P(\text{top secure}) \bullet P(\text{bottom secure})$
Since $P(\text{top secure}) = 1\text{-}\prod_{i=1}^{n} p_{1,i}$ and $P(\text{bottom secure}) = 1\text{-}\prod_{i=1}^{n} p_{2,i}$ we see that
$P(\Sigma \text{ insecure}) = 1\text{-}(1\text{-}\prod_{i=1}^{n} p_{1,i})(1\text{-}\prod_{i=1}^{n} p_{2,i})$
If, we assume as above, that the $p_{j,i}$ are bounded away from one, then
$\lim_{n \rightarrow \infty} \prod_{i=1}^{n} p_{j,i} \rightarrow 0$, and thus $P(\Sigma \text{ insecure}) \rightarrow 0$.

We see that analyzing each circuit on ad hoc basis does not work to our advantage. A formal model for calculating circuit insecurities would provide a useful framework for analyzing both strengths and weaknesses of various protection domain architectures.

4. Formal Model and Discussion

4.1 Formal Theory

DEFINITION: A *circuit* Σ is a connected graph (we refer to the vertices as nodes) with the following properties:

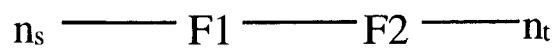
- There are two special nodes: the *start node* n_s and the *terminal node* n_t.
- Each node n_i is an independent Bernoulli random variable and is assigned a probability p_i of being insecure and probability $1\text{-}p_i$ of being secure.
- $p_s = p_t = 1$.

Given an edge e of Σ, we let N(e) denote the nodes associated with e. This is either a set of one element (the corresponding edge is a loop) or two elements.

The interpretation of this definition is that Σ models the various ways insecurities may flow from the source n_s to the sink n_t. Here we apply the analogy with Moore and Shannon's [M1, M2] work on the flow of electricity in a circuit. To be specific, Moore and Shannon examine, without our formalisms, the paths necessary to complete an electrical circuit. Their analysis also applies to our present problem. What

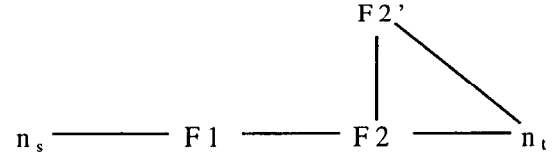
paths (of edges) are necessary to expose all the various realistic insecurities of our layered protection system?

EXAMPLE f1: For example, suppose there are two serial security mechanisms F1 and F2 protecting n_t. The insecurity flows out of n_s and then must go through F1 and F2. Although a path from n_s, to F1, to F2, to F1, to F2, and finally to n_t, is a valid topological path, it would be an inefficient path for a penetrator to take.



Once a penetrator is past F1, it would have no reason to go back to F1. The interpretation is that F1 is the first level of security that is broken, and once it is broken, there would no reason to re-break it.

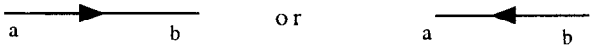
EXAMPLE f2: This example postulates that there are security mechanisms F1, F2, and F2'. F2 serially follows F1, n_t follows both F2 and F2'. However, an edge also connects F2 to F2'.



Suppose an insecurity has traveled from n_s to F1 to F2. Once it passes F2, why would it attempt to go to F2'? This would just decrease its chances of getting to n_t. Once the insecurity is past F2, it has made it to n_t! Of course, one might ask what F2' is doing there in the first place? We realize that we are stating the obvious but we do so to develop a rigorous mathematical model. One advantage of a formal model is that we can easily identify poor system engineering architectures (such as the above), or at least question our decisions for further analysis.

DEFINITION: Given an edge e of Σ, if N(e) has two elements we can assign an *orientation* to e by designating one element of N(e) as ${}^{L}c$, and the other element of N(e) as e^{R}. If N(e) is a singleton, then e has one orientation given by ${}^{L}e = e^{R} = N(e)$. We say that an edge is *oriented* if it has been assigned an orientation.

Example:



We have used the term "path" heuristically above. Now let us give it a precise definition.

DEFINITION: An *oriented path* in Σ, is a tuple of edges, the i^{th} component being edge e_i, along with an orientation for each e_i, such that $e_i^{R} = {}^{L}e_{i+1}$. If the tuple has m components, we say that the oriented path is of length m.

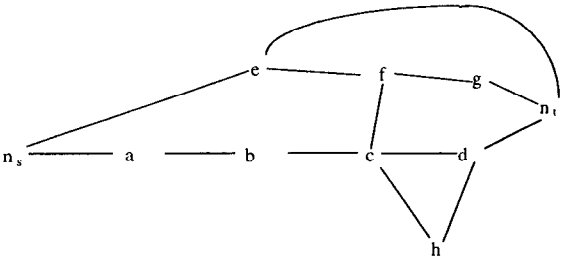
DEFINITION: We say that an oriented path in Σ is *from* node a to node b if ${}^{L}e_1 = a$ and $e_z^{R} = b$, where e_1 (e_z) is the first (last) component of the oriented path.

We designate an oriented path from node a to node b by $\upsilon[a,b]$.

DEFINITION: The *signature* of an oriented path $\upsilon[a,b]$ of length m, is the m+1-tuple $< {}^{L}e_1, e_1^{R}, e_2^{R}, \ldots, e_m^{R} >$ (which is the same as $< a, e_1^{R}, e_2^{R}, \ldots, b >$).

We use the notation $\phi(\upsilon[a,b])$ for the signature, and in the above used <,> instead of {,} for reasons of clarity.

EXAMPLE f3: At this point, we will use the more generic terminology of nodes, instead of security mechanisms/protection domains. We will also continue to switch between the n_i notation and just alphabetic letters to designate the distinct nodes.



Consider (a subset of) the oriented paths from n_s to n_t (with signatures) $<n_s,a,b,c,d,n_t>$, $<n_s,e,f,g,n_t>$, $<n_s,e,n_t>$, $<n_s,a,b,c,f,g,n_t>$, $<n_s,a,b,c,f,e,n_t>$, and $<n_s,a,b,c,h,d,n_t>$. The first oriented path corresponds to a layering of protection domains a, then b, then c, and the last d. The last oriented path is unrealistic; once the insecurity has flowed through c, there is no reason to take a detour through h (because it would have to go through d to get at n_t!). Therefore, we must throw out the last oriented path. The third oriented path would provide an efficient way for a penetrator to attempt to get to n_t. Seeing such an oriented path would cause one to ask "Why is the system designed so that a penetrator can bypass the protection domains f and g if you have made it past e?" This causes us to also throw the second oriented path out as a way of intruding into n_t---there is no reason to follow the second oriented path if you have already gotten through e. Consider the fourth and fifth oriented paths. The fourth is a possible means of

intrusion if you start out on the first oriented path and are unable to get by d. For that matter, if one has gotten through a,b,c and f, but are unable to get through g, one could go through e instead, and hence the fifth oriented path seems reasonable. However, it is not reasonable. Using the fifth path one still has to go through node e after node f. Therefore a smart penetrator would use $<n_s,e,n_t>$ instead. Thus, our analysis leads to the following definitions.

DEFINITION: Consider the set of all oriented paths in Σ from n_s to n_t . From this, form the finite subset made up of those oriented paths that never repeat an edge. We call this set G.

G has been defined in this way to take, oriented paths from n_s to n_t that do not represent a valid attack path (we do not want cycles between nodes), out of consideration. Unfortunately, this set G still contains unrealistic attack paths (those that unnecessarily branch out).

Begin Process: If $g \in$ G (so g is of the form $\upsilon[n_s,n_t]$) we consider the j-tuple $\phi(g)$. If the (j–k)-tuple obtained from $\phi(g)$ by deleting k components is in fact the signature of another element of G we form the set G–$\{g\}$. End Process

Repeat this process with G replaced by G–$\{g\}$ until there are no more elements to delete from G. The set obtained by exhaustively performing this process is called the set of *penetrating flows* in Σ, we let Ω denote the set of penetrating flows. We abuse notation and denote the set of signatures for all penetrating flows by $\phi(\Omega)$.

Thus, a penetrating flow is simply an oriented path from n_s to n_t satisfying a minimality condition that represents a valid attack path. We do not consider the complexity of this process. Of course, if one were to use this in practice on a large network such things should be considered. We do not feel that this process/algorithm can be derived from standard minimal path algorithms because it is not a minimization problem alone. For example in the 2P-circuit both paths are "counted".

DEFINITION: Given $g \in$ G, there is a unique penetrating flow in Ω corresponding to g under the above process. We refer to this as the *penetrating representative* of g, and denote it by $[g]$.

NOTE:

1. No node is duplicated in the signature of a penetrating flow.
2. We have an algebraic flavor to our model---we may view Ω as the coset space of G under the action of deletion. That is, G maps into Ω via deletion and a coset representative is signified by $[\gamma]$.

$$G/\text{deletion} = \Omega, \text{ and } g \rightarrow [g]$$

Further, we see that G = Ω iff there are no unnecessary (in terms of what is required for a realistic security breach) oriented paths from n_s to n_t. Hence, this can be taken as the mark of good security architecture.

Consider Example f3.
The set of penetrating flows is

Ω = { $<n_s,a,b,c,d,n_t>$, $<n_s,e,n_t>$, $<n_s,a,b,c,f,g,n_t>$}.

The oriented paths $<n_s,e,f,g,n_t>$, $<n_s,a,b,c,f,e,n_t>$ are taken out of G because of $<n_s,e,n_t>$, and $<n_s,a,b,c,h,d,n_t>$ is removed from G because of $<n_s,a,b,c,d,n_t>$.

The set Ω associated with Σ represents ways that an insecurity could realistically attempt to flow from n_s to n_t. The other oriented paths in G would also let an insecurity flow from n_t to n_s but they would involve unnecessary nodes; hence, we do not consider them. Note that we have not considered the node probabilities yet. Certain oriented paths in Ω will result in much smaller probabilities of penetration to n_t from n_s than others. However, we must consider all of the elements of Ω when it comes to assigning a probability to Σ being insecure.

At this point, we wish to carefully consider the physical intuition behind our definitions. Assume that g $\in$ G; if we think of n_s as the source of an electrical current, we wish to examine if the current can flow to n_t (there is no resistance along the oriented path). The nodes behave as gates; they are open with probability p_i and closed with probability $1-p_i$. If we examine g, there may be nodes that do not need to be open for the current to flow. Therefore, one does not need to take account of the probabilities of those nodes when assigning a probability of a security flaw flowing along g. These are the nodes that are deleted when we obtain the penetrating representative of g. We see that we only need to know the probabilities of the nodes being open in the penetrating representative of g. Since the probabilities of the nodes of [g] are independent we see that the probability that current flows thorough [g], and hence g , is just the product of the node probabilities of [g].

By using the fact that the nodes are all independent Bernoulli random variables we may (in the standard manner) generate a σ-algebra, and probability measure

upon that σ-algebra, so that the following are well defined.

DEFINITION: Given any collection of nodes N of Σ, we define the event, *N insecure*, as the event that every node in N is insecure, n_s is insecure, n_t is insecure, and any other node is either secure or insecure.

The complement of, N insecure, is N *secure*; i.e., at least one node in N is secure.

Of course, if a node is not in N, except for n_s and n_t, its value is arbitrary with respect to the event of N being insecure. Therefore, the probability of, N insecure, denoted P(N insecure), is simply the product of the node probabilities of the nodes in N. This is due to our independence assumption.

DEFINITION: Given any $\gamma \in \Omega$, we define the event, γ *insecure*, to be the event that the set of nodes composing $\phi(\gamma)$ is insecure.

This is well defined since no nodes are duplicated in $\phi(\gamma)$. The complement of, γ insecure, is, γ *secure*, (there is at least one node in γ that is secure). If a penetrating flow is secure, an insecurity cannot flow from the start to the terminal node, and the penetrating flow is "safe".

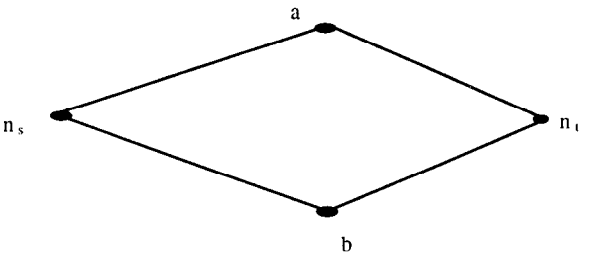
We may now view Σ itself in a probabilistic sense and discuss the event that Σ is insecure. By this, we mean that at least one of the penetrating flows is insecure. To be precise:

DEFINITION: We define the event, Σ *insecure*, by
Σ insecure $= \cup_{\gamma\in\Omega}(\gamma \text{ insecure}) = \{\exists\gamma \in \Omega$, such that γ insecure$\}$.

The complement of the event, Σ insecure, is the event, Σ *secure*, which is the event that every penetrating flow is secure. To be precise
Σ secure $= \cap_{\gamma\in\Omega}(\gamma \text{ secure}) = \{\forall\gamma \in \Omega, \gamma \text{ secure}\}$

Geometrically, penetrating flows γ are, aside from n_s and n_t (which they all share), either disjoint or have nodes in common. However, even if they do not share nodes, the events, γ insecure, cannot be considered disjoint. We will illustrate this in the following example.

Example f4:



The set of penetrating flows consists of the following two oriented paths:
1= $<n_s,a,n_t>$ and 2= $<n_s,b,n_t>$. P(1 insecure) = p_a, P(2 insecure) = p_b, but
P(1 insecure $\cap$ 2 insecure) = $p_a p_b$. The reason for this is that the event, 1 insecure, can be written as the disjoint union of two sample space elements:
1 insecure = {all nodes insecure} $\cup$ {all nodes except node b are insecure}
and similarly:
2 insecure = {all nodes insecure} $\cup$ {all nodes except node a are insecure}
Therefore, the intersection of the two events is the sample space element
{all nodes insecure} which has a probability of $1 \bullet p_a p_b \bullet 1$. Thus,
P(1 insecure $\cup$ 2 insecure) = P(1 insecure) + P(2 insecure) - P(1 insecure $\cap$ 2 insecure)
$= p_a + p_b - p_a p_b$

Example f5: Consider the following more complicated circuit:

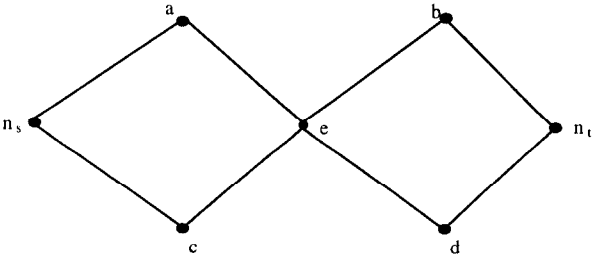


Figure 6: A complicated circuit

The set Ω consists of four oriented paths (which we give by their signatures):
$\Omega = \{1,2,3,4\}$
$1 = <n_s,a,e,b,n_t>$
$2= <n_s,c,e,d,n_t>$
$3= <n_s,a,e,d,n_t>$
$4= <n_s,c,e,b,n_t>$

$P(\Sigma \text{ insecure}) = P(\cup_{\gamma \in \Omega}\{\gamma \text{ insecure}\})$
$= P(1 \text{ insecure} \cup 2 \text{ insecure} \cup 3 \text{ insecure} \cup 4 \text{ insecure})$
$= P(1 \text{ insecure}) + P(2 \text{ insecure}) + P(3 \text{ insecure})$
$+ P(4 \text{ insecure}) - P(1 \text{ insecure}, 2 \text{ insecure})$
$- P(1 \text{ insecure}, 3 \text{ insecure}) - P(1 \text{ insecure}, 4 \text{ insecure})$
$- P(2 \text{ insecure}, 3 \text{ insecure}) - P(2 \text{ insecure}, 4 \text{ insecure})$
$- P(3 \text{ insecure}, 4 \text{ insecure})$
$+ P(1 \text{ insecure}, 2 \text{ insecure}, 3 \text{ insecure})$
$+ P(1 \text{ insecure}, 2 \text{ insecure}, 4 \text{ insecure})$
$+ P(1 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure})$
$+ P(2 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure})$
$- P(1 \text{ insecure}, 2 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure})$

Let us calculate the various terms:
$P(1 \text{ insecure}) = 1 \cdot p_a p_e p_b \cdot 1 = p_a p_e p_b$
$P(2 \text{ insecure}) = p_c p_e p_d$
$P(3 \text{ insecure}) = p_a p_e p_d$
$P(4 \text{ insecure}) = p_c p_e p_b$
$P(1 \text{ insecure}, 2 \text{ insecure}) = p_a p_b p_c p_d p_e$. This is because the intersection of the two events is the event that every node in the circuit is insecure.
$P(1 \text{ insecure}, 3 \text{ insecure}) = p_a p_b p_d p_e$, since node c being insecure is not in the intersection. By this, we mean that node c can be either insecure or secure. Hence, p_c does not enter into the calculation.
$P(1 \text{ insecure}, 4 \text{ insecure}) = p_a p_b p_c p_e$
$P(2 \text{ insecure}, 3 \text{ insecure}) = p_a p_c p_d p_e$
$P(2 \text{ insecure}, 4 \text{ insecure}) = p_b p_c p_d p_e$
$P(3 \text{ insecure}, 4 \text{ insecure}) = p_a p_b p_c p_d p_e$
$P(1 \text{ insecure}, 2 \text{ insecure}, 3 \text{ insecure}) = P(1 \text{ insecure}, 2 \text{ insecure}, 4 \text{ insecure})$
$= P(1 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure}) = P(2 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure})$
$= p_a p_b p_c p_d p_e$, since, in the triple intersections, all nodes must be insecure.
and finally, $P(1 \text{ insecure}, 2 \text{ insecure}, 3 \text{ insecure}, 4 \text{ insecure}) = p_a p_b p_c p_d p_e$.
Thus,

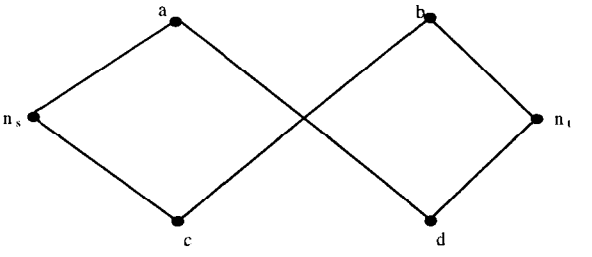
$$P(\Sigma \text{ insecure}) = p_a p_b p_e + p_a p_d p_e + p_b p_c p_e + p_c p_d p_e - p_a p_b p_d p_e - p_a p_b p_c p_e - p_a p_c p_d p_e - p_b p_c p_d p_e + p_a p_b p_c p_d p_e .$$

This was a lot of work! What happens if we want to put some extra nodes into the circuit and then analyze the circuit insecurity after this? We saw in the previous section how introducing extra nodes can affect circuit insecurities. What happens if we have a very complex circuit? We need an easier way to calculate circuit insecurities and to analyze modifications to the security architecture.

4.2 Reduction Formulas

In this subsection, we extrapolate some of the "reduction" type rules put forth in [M1, M2] to the concept of insecurity flow. Our reduction formulas are not meant to be exhaustive. These rules may fall short when the circuit topology in no longer *faithfully* represented in 2-dimensions (e.g., nodes of a cube). We view these reductions as a first step towards a full analysis of generalized complex circuits. However, we do view series and parallel decomposition along bridging nodes to be an important technique and present it in this subsection. More complicated topologies may be analyzed and used as building blocks for further reduction formulas. Note that any circuit may be probabilistically analyzed by the "long hand" methods presented in the previous subsection.

If the node e is missing from the above circuit in figure 6 (equivalent to node e being totally insecure, that is $p_e = 1$), we have a much simpler circuit Σ^{-e}.



The middle crossing designates that there are edges from a to b, a to d, c to b, and c to d. We will use the following "curved heuristic" in our circuit diagrams to make this clearer.

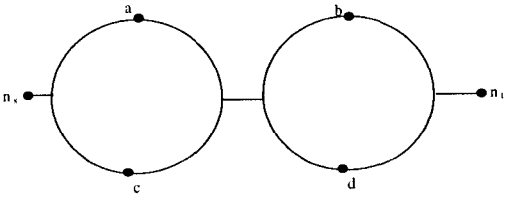


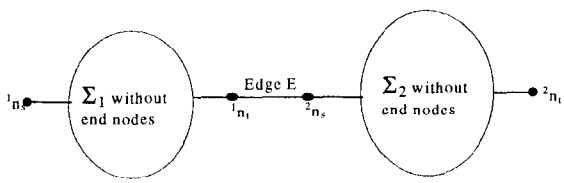
Figure 7: A simplified circuit

The left-hand circle is just node a and node c in parallel, flowing into a node that is always insecure (the "missing" node e), whereas the right-hand circle has node b and node d in parallel. The two circles are glued in series. We see that three important steps are occurring.

1- Get rid of the central node.
2- Use a parallel composition rule.
3- Use a serial composition rule.

Let us make this precise.

DEFINITION: Given circuits Σ_1 with edges ${}^1e_\alpha$ and nodes ${}^1n_i$, and Σ_2 with edges ${}^2e_\beta$ and nodes ${}^2n_j$, we may form the *serial composition* of these two circuits, written $\Sigma_1 \text{and}_2$, by forming a new circuit from ${}^1e_\alpha$, ${}^1n_i$, ${}^2e_\beta$, and ${}^2n_j$ along with a
new edge E with orientation such that ${}^L E = {}^1n_t$ and $E^R = {}^2n_s$



Consider the set of penetrating flows for $\Sigma_1 \text{and}_2$. If $\gamma \in \Omega_1 \text{and}_2$, then the tuple $\phi(\gamma)$ must contain the two consecutive components ${}^1n_t$ and ${}^2n_s$, respectfully. In fact, $\phi(\gamma)$ is the tuple formed by a valid signature of a penetrating flow from Ω_1 being the first part of the tuple and the valid signature of a penetrating flow from Ω_2 being the second (and last) part of the tuple and visa versa. In other words, $\gamma \in \Omega_1 \text{and}_2$ iff its signature is of the form

$$\phi(\gamma) = <{}^1n_s,\ldots, {}^1n_t, {}^2n_s,\ldots, {}^2n_t>,$$

where $<{}^1n_s,\ldots, {}^1n_t>$ is the signature of a penetrating flow from $\gamma 1$ from Σ_1, and
$<{}^2n_s,\ldots, {}^2n_t>$ is the signature of a penetrating flow $\gamma 2$ from Σ_2. Hence,

$$P(\gamma \text{ insecure}) = P(\gamma 1 \text{ insecure}) \bullet P(\gamma 2 \text{ insecure})$$

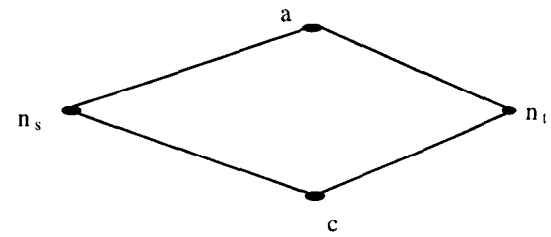
by the node independence condition. The same holds for all of the penetrating paths.
What about the circuit insecurity?
$P(\Sigma_1 \text{and}_2 \text{ insecure})$ = P(at least one element of $\Omega_1 \text{and}_2$ is insecure). However, from above we see that this is the same as
P(at least one element of Ω_1 is insecure $\cap$ at least one element of Ω_2 is insecure).
Since Σ_1 and Σ_2 share no nodes the events, at least one element of Ω_1 is insecure, and, at least one element of Ω_2 is insecure, are independent. Therefore,
$P(\Sigma_1 \text{and}_2 \text{ insecure})$
= P(at least element of Ω_1 is insecure) • P(at least element of Ω_2 is insecure)
= $P(\Sigma_1 \text{ insecure}) \bullet P(\Sigma_2 \text{ insecure})$

FORMULA:
$P(\Sigma_1 \text{and}_2 \text{ insecure}) = P(\Sigma_1 \text{ insecure}) \cdot P(\Sigma_2 \text{ insecure})$

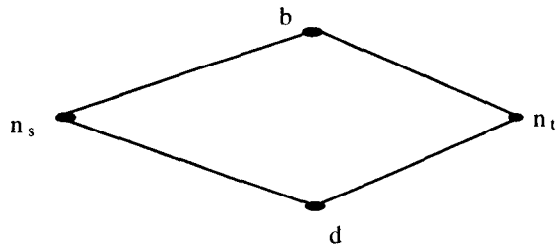
DEFINITION: Given a circuit Σ we say it is *serial decomposable to* $\Sigma_1 \text{and}_2$, written,

$\Sigma \sim \Sigma_1 \text{and}_2$, if there are circuits Σ_1 and Σ_2, such that if we take the set $\phi(\Omega_1 \text{and}_2)$ and delete the components ${}^1n_t$ and ${}^2n_s$ from the signatures, we are then left with the set $\phi(\Omega)$. (It is understood that start and terminal nodes are equivalent, and node probabilities are the same.)

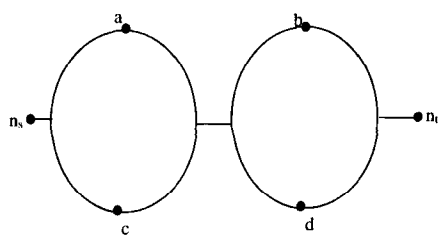
To illustrate this let Σ_1 be the circuit:



and let Σ_2 be the circuit:



Let Σ be the circuit:



We see that the four signatures of Ω, are the same as the four signatures of $\Omega_1 \text{and}_2$ with the middle start and terminal nodes removed and the appropriate correspondence of the end nodes.

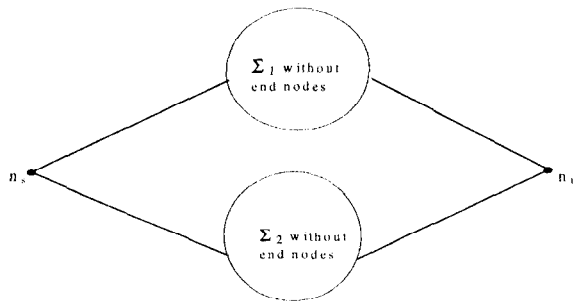
Since start and terminal node probabilities are 1, we see that circuit insecurity values are the same for the circuit and its serial decomposition.

Therefore, if a circuit is serial decomposable, we have an "easier" way to calculate the probability of it being insecure. That is if $\Sigma \sim \Sigma_1 \text{and}_2$, then

$$P(\Sigma \text{ insecure}) = P(\Sigma_1 \text{and}_2 \text{ insecure})$$
$$= P(\Sigma_1 \text{ insecure}) \bullet P(\Sigma_2 \text{ insecure})$$

DEFINITION: Given circuits Σ_1 with edges ${}^1e_\alpha$ and nodes ${}^1n_i$, and Σ_2 with edges ${}^2e_\beta$ and nodes ${}^2n_j$, we may form the *parallel composition* of these two circuits, written $\Sigma_1 \mathrm{or}_2$, by forming a new circuit from ${}^1e_\alpha$, ${}^1n_i$, ${}^2e_\beta$, and ${}^2n_j$ by taking ${}^1n_s = {}^2n_s$, and ${}^1n_t = {}^2n_t$.

The following heuristically represents $\Sigma_1 \mathrm{or}_2$.



Now we wish to calculate $P(\Sigma_1 \mathrm{or}_2$ insecure). If γ is a penetrating flow of $\Sigma_1 \mathrm{or}_2$, it is either (translating n_s back into ${}^1n_s$ or ${}^2n_s$ and similarly for n_t) in Ω_1, or Ω_2, (but not both). We will calculate the complementary probability, $P(\Sigma_1 \mathrm{or}_2$ secure).

That is, what is the probability that every penetrating flow is secure? By our above discussion, we see that

$P(\Sigma_1 \mathrm{or}_2$ secure)

= P(every penetrating flow in Σ_1 is secure $\cap$ every penetrating flow in Σ_2 is secure).

Since the two circuits share no nodes (do not consider start and terminal nodes),

we are taking the intersection of two independent events. Hence,

$P(\Sigma_1 \mathrm{or}_2$ secure)

= P(every penetrating flow in Σ_1 is secure)

• P(every penetrating flow in Σ_2 is secure)

= $P(\Sigma_1$ secure) • $P(\Sigma_2$ secure).

Since for any Σ, $P(\Sigma$ insecure) = 1- $P(\Sigma$ secure), we have

$P(\Sigma_1 \mathrm{or}_2$ insecure) = 1- (1- $P(\Sigma_1$ insecure))(1- $P(\Sigma_2$ insecure))

By using induction we may obviously define, $\Sigma_1 \mathrm{or}_2 \mathrm{or}_{\ldots n}$ and $\Sigma_1 \mathrm{and}_2 \mathrm{and}\ldots_n$, and obtain (note figures 7&8 in [M1].)

$$P(\Sigma_1 \mathrm{or}_2 \mathrm{or}_{\ldots n} \text{ insecure}) = 1 - \Pi_i (1 - P(\Sigma_i \text{ insecure}))$$
$$P(\Sigma_1 \mathrm{and}_2 \mathrm{and}\ldots_n \text{ insecure}) = \Pi_i P(\Sigma_i \text{ insecure})$$

We can also extend serial decomposability in this manner. Note we do not discuss parallel decomposition, unlike serial decomposition, because no extra nodes are introduced in parallel composition. Hence, we can just say that a circuit *is* the parallel composition of two simpler circuits.

These formulas make precise the probability calculations from section 2.

DEFINITION: We say that node β of Σ is a *bridge* if there are two signatures of penetrating flows that have β as a component with different nodes as the next components.

In other words, with respect to penetrating flows, there are two or more valid ways for an insecurity to flow once an insecurity has breached node β. We wish to see what happens to the insecurity properties of Σ if β is always insecure or always secure. By conditioning we have

$P(\Sigma$ insecure)

= $P(\Sigma$ insecure | β insecure) • $P(\beta$ insecure) + $P(\Sigma$ insecure | β secure) • (1- $P(\beta$ insecure))

Thus, $P(\Sigma$ insecure) = p_β • $P(\Sigma$ insecure | β insecure) + $(1\text{-}p_\beta)$ • $P(\Sigma$ insecure | β secure)

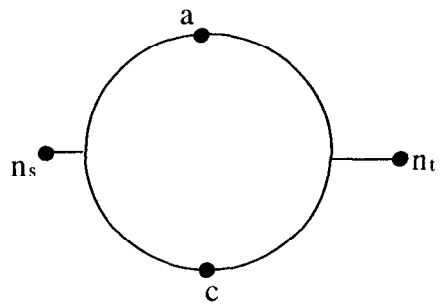
This formula can be very useful when calculating the probabilities (as we shall show). Note the similarity with [Eq. 3, M1].

Let us return to example f5, we see that node e is a bridge. So,

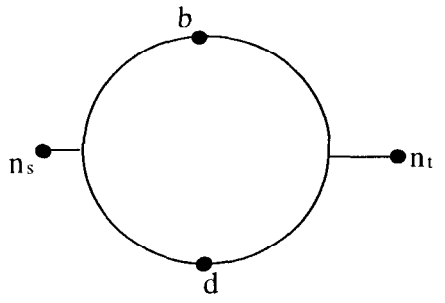
$P(\Sigma$ insecure) = p_e • $P(\Sigma$ insecure | β insecure) + $(1\text{-}p_e)$ • $P(\Sigma$ insecure | β secure)

The event (Σ insecure | β insecure) is the event that circuit Σ^{-e} (as given in figure 7) is insecure. The circuit Σ^{-e} is serial decomposable into $\Sigma_{1 \mathrm{and}\ 2}$,

where Σ_1 is

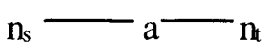


and Σ_2 is

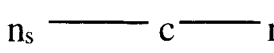


We can further simplify both Σ_1 and Σ_2 since Σ_1 is the parallel composition $\Sigma_{a \text{ or } c}$, and Σ_2 is the parallel composition $\Sigma_{b \text{ or } d}$. Where

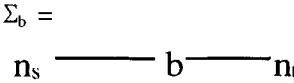
Σ_a =



Σ_c =



Σ_b =



Σ_d =

Therefore,
$P(\Sigma_a \text{ insecure}) = p_a$
$P(\Sigma_c \text{ insecure}) = p_c$
$P(\Sigma_b \text{ insecure}) = p_b$
$P(\Sigma_d \text{ insecure}) = p_d$
So, $P(\Sigma_1 \text{ insecure}) = 1-(1- p_a)(1- p_c)$, and
$P(\Sigma_2 \text{ insecure}) = 1-(1- p_b)(1- p_d)$.
Therefore, $P(\Sigma_{1 \text{and } 2} \text{ insecure})$

$$= [1-(1- p_a)(1- p_c)] \bullet [1-(1- p_b)(1- p_d)]$$
$$= [p_a + p_c - p_a p_c] \bullet [p_b + p_d - p_b p_d]$$

The event (Σ insecure | β secure) = (Σ insecure | e secure) = $\varnothing$ since e is in the signature of every penetrating flow of Σ. Hence,

$$P(\Sigma \text{ insecure}) = p_e \bullet [p_a + p_c - p_a p_c] [p_b + p_d - p_b p_d] + (1-p_e) \bullet 0$$
$$= p_a p_b p_e + p_a p_d p_e + p_b p_c p_e + p_c p_d p_e - p_a p_b p_d p_e - p_a p_b p_c p_e - p_a p_c p_d p_e - p_b p_c p_d p_e + p_a p_b p_c p_d p_e .$$

This agrees with our previous result and requires much less work.

Let us summarize our techniques for calculating circuit insecurities.

Probabilistically condition on a bridge node being insecure or secure, and then attempt to decompose, or view as, serial or parallel circuits.

We will not repeat the examples of adding security mechanisms as in the previous section (for the informal model). Those same ideas and techniques hold for the formal model and show how system designers can add nodes to a circuit to increase the security to a desired level. However, we stress that the formal model described in this section facilitates the analysis of such techniques. The strength of our formalism is that it separates the "wheat from the chaff" and provides a framework for focusing on the significant parts of the problem. Further, our method demonstrates how to reduce complex calculations to relatively simple parallel and serial constructs, and then apply our formulas to those simpler constructs.

5. Conclusions and Future Work

Given today's open, distributed, and complex computer systems, security is both increasingly critical and hard to attain. The more complex computer systems become, the harder the analysis of the security vulnerability of the system becomes. Ad hoc methods of analyzing security vulnerability have limitations. We have put forth a new paradigm of insecurity flow and quantified its security via a formal mathematical model. Our paradigm is useful to analyze the security vulnerability even before the security measures are deployed. It specifically helps the security designer to

- find the security vulnerability of the system
- find redundant or useless security measures
- find efficient ways to add security mechanisms.

This paradigm also allows the security architect to design security measures based on security vulnerability versus cost analysis.

When the system is too complex, the model associated with our paradigm allows a decomposition into simpler system models for ease of analysis. The analysis results of simpler models can be combined to analyze the security vulnerability of the complex model as a whole.

If the community finds our work of interest, we may

- extend our thinking beyond Bernoulli percolation type models,
- extend our paradigm of insecurity flow to incorporate dependencies among the protection domains,
- investigate critical probabilities that would insure an insecure system, and
- investigate if further concepts in dynamical systems can be applied to security.

We also wish to contrast our work against the complementary graph-theoretical work in [D1], [D2], and [O] where arcs represent security values, and nodes represent the security-related subjects of a particular system. In addition, further reduction formulas need to be developed to deal with more complex circuit topologies.

6. Acknowledgments

We especially thank Mara Moss for her helpful discussions of the robustness of point of sales transactions, which motivated this paper, along with her insightful comments on our research in general.

We also thank Earl Boebert, LiWu Chang, Yves Deswarte, Carl Landwehr, Cathy Meadows, Bruce Montrose, and Rodolphe Ortalo for their helpful comments, observations, and comments on earlier versions of the paper. We also appreciate comments of the anonymous referees and participants of the workshop. Research partially supported by the Office of Naval Research.

7. References

[B] W. E. Boebert and R. Y. Kain, "A Practical Alternative to Hierarchical Integrity Policies," Proc. 8th National Computer Security Conference, Gaithersburg, MD, 1985.

[C] OMG, "CORBA Security," version 1.1 OMG document Numbers 96-08-03 through 96-08-06, 1996.

[D1] M. Dacier, "Towards Quantitative Evaluation of Computer Security," Doctoral Thesis, LAAS 94488 (in French), Institut National Polytechnique de Toulouse, Dec.1994.

[D2] M. Dacier, Y. Deswarte, and M. Kaâniche, "Quantitative Assessment of Operational Security: Models and Tools," LAAS Research Report 96493, May 1996 (Extended version of "Models and Tools for Quantitative Assessment of Operational Security," Proc. IFIP/SEC'96).

[K] B.O. Koopman, "Search and Screening," Pergamon Press, Inc, New York, 1980.

[La] P.M. Lam, "A Percolation Approach to the Kauffman Model," Journal of Statistical Physics, Vol. 50, Nos. 5/6, pp. 1263-1269, 1988.

[LB] B. Littlewood, S. Brocklehurst, N. Fenton, P. Mellor, S. Page, and D. Wright, "Towards Operational Measures of Computer Security," Journal of Computer Security, 2, pp. 211-229, 1993.

[Le] T.M.P. Lee. "Statistical Models of Trust: TCB's vs. People," Proceedings of 1989 IEEE Symposium on Security and Privacy, IEEE CS Press, 1989.

[Mc] J. McLean, "A General Theory of Composition for Trace Sets Closed Under Selective Interleaving Functions," Proceedings of 1994 IEEE Symposium on Research in Security and Privacy, IEEE Press, 1994.

[Mi] J.K. Millen, "Covert Channel Capacity," Proceedings of 1987 IEEE Symposium on Security and Privacy, pp. 60-73, IEEE CS Press, 1987.

[M1] E.F. Moore and C.E. Shannon, "Reliable Circuits Using Less Reliable Relays I," *Journal Franklin Institute*, Vol. 262, pp.191-208, Sept., 1956.

[M2] E.F. Moore and C.E. Shannon, "Reliable Circuits Using Less Reliable Relays II," *Journal Franklin Institute*, Vol. 262, pp.281-297, Oct., 1956.

[O] R. Ortalo, Y. Deswarte, M. Kaâniche, "Experimenting with Quantitative Evaluation Tools for Monitoring Operational Security," Proc. 6th International Working Conference of Dependable Computing for Critical Applications, Grainau, Germany, March, 1997.

[W] J.C. Wierman, "Percolation Theory," The Annals of Probability, Vol. 10, No. 3, pp. 509-524, 1982.