

TOWARD A MODEL OF SECURITY FOR A NETWORK OF COMPUTERS

William H. Murray
Deloitte & Touche
21 Locust Avenue, Suite 2D
New Canaan, Connecticut 06840
WHMurray@DOCKMASTER.NCSC.MIL

Patrick Farrell
Department of Computer Science
George Mason University
Fairfax, Virginia, 22030-4444
pfarrell@cs.gmu.edu

ABSTRACT

This paper proposes a model for evaluating the security of networks of computers. It is about the security of the collection of systems connected, rather than about security of the connecting infrastructure. It proposes that: 1) the networks of interest can be modelled as a collection of abstract single-user single-task machines connected at their user interfaces; 2) the security of such a net can be evaluated by comparing the cost of a successful attack to its value; 3) the cost of attack can be evaluated in terms of the work effort to the attacker and that the value can be evaluated by calculating the how much the success reduces the cost of subsequent attacks. Illustrations are given of how success against one node reduces the cost of attack against other nodes.

The paper suggests uses for the proposed model. It recommends network security practices that are suggested by the proposed model. It also suggests additional areas of research.

INTRODUCTION

To say that modern networks of computers are complex and dynamic is to restate the obvious. It is difficult to make statements or answer questions about any property of such networks; even such seemingly simple questions as size and scope resist easy answers. Nonetheless, mathematical modelling enables us to make useful, predictive, though not absolute, statements about even more complex systems, such as the economy or the weather.

PURPOSE

In a paper presented to the National Conference on Computing and Values, Murray¹ asserted that we need to be able to make statements about the security of populations and networks of computers as well as about individual computers. He suggested that we need to be able to answer questions such as what happens to the security of two systems when they are joined together? What happens to the security of a system when it is joined to a network? What happens to the security of the network?

The ideas presented here are intended to help us answer such questions. They are intended as a start toward a mathematical model of the security of a network of computers.

APPROACH

We begin with an abstract atomic system that is so simple that we can make statements about its security. This system is simpler than the kind of multi-user system that we normally use to illustrate computer security. It is intended to abstract a set of qualities and characteristics that will be relevant to its role in a network while putting aside such otherwise significant characteristics as application and environment which are obscured by its membership in a large population of dissimilar systems.

We are looking for properties that are fractal and composable, i.e., independent of scale, as true of the whole as they are of the parts. We are looking for properties that can be used to describe the security of

otherwise dissimilar systems. We are looking for properties that are independent of system type, management, or quality of implementation.

CONTRAST TO OTHER APPROACHES

We take note of Courtney's first law: "Nothing can be said about the security of a system except in the context of an application or environment." However, for our purposes, we assert that it is the generality and flexibility of the system that is of interest. For purposes of being able to talk about the network, we focus on those properties which are shared across systems, and ignore or conceal those that are unique or peculiar to a few.

This concept of security can be contrasted to that dealt with in the *Orange Book*². The *Orange Book* wishes to be able to make absolute statements about single systems; we wish to be able to compare effects within populations of systems. The *Orange Book* describes levels of security in terms of policies and the ability to enforce them in terms of functions or capabilities. Our approach is more like that of the cryptographer, in which the assumption is that any code can be broken at some cost. We wish to be able to make statements about relative cost. That is, we wish to make statements about measures that will increase or decrease the cost of attack or the value of success.

CAUTION TO THE READER

This paper is directed at the collective security of computers networked together. It concentrates on the security of the collection as a whole, with recognition of the contribution towards collective security provided by each abstract machine. This paper is not about telecommunications or the connecting infrastructure (network), nor is it about computer security in the traditional sense. It is not about system security but about the security of networks of security.

THE SIMPLE MODEL

In this section we begin to introduce the primitive components of the model.

THE ABSTRACT MACHINE

Let M be an atomic single-user single-tasking single-service abstract machine of such strength and isolation that the only path of attack is through its user interface. M contains a secret, privilege, or resource, R , of value, V , and cost of loss or compromise, L .

Un-privileged user processes within a multi-user system would be modelled as a collection of such abstract machines. The privileged processes on such a machine would be modelled as containing, as resources, the entire collection.

SECURITY MODEL

Let C_a be the cost of an attack trial and N the expected number of trials required for success.

We define the abstract machine to be *attack secure* if the cost of a successful attack, $C_a N$ is greater than V . That is, a rational attacker will not attack the system if the perceived cost of attack is greater than the value of success. As we show, this cost can be explicitly modeled.

Within successful attacks, we distinguish between *penetrations* and *breaks*. We would consider an abstract machine to be *penetrated* if the cost of attack fell below the value of successful attack and for the length of time that such condition persisted. We consider the machine to be *broken* if the condition would not remedy itself automatically. Thus, if the attacker were to obtain the password to a machine, the machine would be considered penetrated for the life of the password. If access to the machine included the

ability to change the password, and if the attacker changes it, then the machine is broken. The importance of this distinction can be illustrated by the following comparison. When my personal computer is in auto-answer mode, it is protected by a seven digit one-time password. It has a very high cost of attack. However, a successful attack includes the ability to insert a secret door in the one-time password mechanism, that is to break the machine. While it might require some special knowledge, and might be difficult to accomplish in a limited time, the privilege to do it is inherent in the penetration. The situation may not remedy itself automatically and the attacker can continue to use the system at reduced cost.

Contrast this to an unprivileged user ID on a Multics system using the same one-time password mechanism. A successful penetration, i.e., a one-in-ten-million guess, does not include the privilege of inserting a secret door. The abstract machine does not include the ability to alter its own logon security. Once the session is ended, the cost of attack returns to the old level automatically. The abstract machine was penetrated, but it was not broken.

While this definition of security may appear to be absolute, it is in fact, relative. We may never know all of the parameters of the model in the manner necessary to make absolute statements about the security of a particular machine. However, we may know enough to make comparative statements about two machines.

This scalar value can be replaced with a time series. Thus, the security evaluation term, $C_a N = V$ becomes $C_a(t_i)N(t_i) = V(t_i)$ at the time t_i . (While this is not important when evaluating the security of a single abstract machine, it will become extremely important when we attempt to evaluate a network.)

ASSUMPTIONS

We note the following simplifying assumptions:

- The model is limited to things that can happen at the user interface. The user interface is that which any user sees when he approaches the system. It includes all of the services that one can get with, or without, logging on. It includes even those services that are normally reserved to privileged users if they are available at the same interface. We ignore things that might happen at a privileged or local interface that would not be visible from the network.
- An attack is an attempt to obtain any service to which one is not normally authorized. It will usually include one or more attempts to logon to the system. Each trial, or associated group of trials may be modelled, examined, or evaluated independently.
- R is the target of attack, the product of successful attack, and that which the owner desires to protect. R may be a secret, such as intelligence or a password; a privilege or a capability, such as the ability to modify a program or place a paid phone call; it may be a connection or a path to a nearby system; or it may be computing capacity, such as might be used to attack a nearby system. For our purposes it is monolithic. Since M is single state, then if the attacker has any part of R, he is assumed to have all of it.
- V and L are related but not necessarily equal. The value of R to an attacker may not be the same as the cost of loss to its owner.
- The cost of access to logon is assumed to be part of the cost of attack, constant, and the same for all potential attackers. Since this is not true in the real world, the model will have to be elaborated to show the components of this cost, and its distribution among attackers.
- The number of trials for success, e.g., half the size of the logon space, is known and constant; again, the model can be elaborated to adjust for uncertainty here. Likewise, the size of the space can be increased, within bounds, to compensate for increases in the value of V. However, all other things being equal, the bounds on the size of the space will ultimately impose an upper bound on the value

of V . That is to say, since there are upper bounds to the size of a password, there are effective bounds on the size of V .

- The number of targets is limited to the single system, M , with value, V . This limit will be relaxed by dividing M , adding it to a population, or by joining it to other systems to form a network.
- We measure work in only in time. The value that the attacker places on his time is not equal for all attackers and not available to us. Neither is it necessary for answering the questions that we want to answer.
- The choice of a single user system is used for simplification. The model will be elaborated to account for the effect of multiple users
- A single task machine was chosen to make logon the only security mechanism of relevance. That is, a compromise of logon will result in a total loss of V ; no other compromise is necessary. This choice was made to illustrate the relationship between the cost of attack and the resource. The elaboration of the model will have to account for multi-user systems with multiple alternative paths of attack to R , and for more complex costs of attack. Conversely, the model will have to be able to account for protective security based upon the division of R across compartments within M .

These assumptions and constraints illustrate some of the dimensions that must be included in a fully elaboration of the model.

THE COST OF ATTACK

The model of the cost of attack will be modelled, in part, by modelling the cost of individual trials. This cost will include the cost to send a unit of coded data times the number of units in the trial, plus the time required to receive and evaluate the response. Thus, for example, the cost of sending six characters is higher than that of sending five, the time required to send it at 300 baud greater than that of sending it at 2400 baud.

We have defined the cost of attack as the cost of a trial times the average number of trials required for success. In its simplest form, $C_a := C_t E[n_a]$.

The cost of a trial is a function of the resources available to the attacker, knowledge, coded information, and capacity. It is related to the protective measures in place. That is, a change in the protective measures may alter either the cost of the trial or the expectation of success of the trial.

If the probability of the success of a trial were to remain constant for the duration of the attack, then a simple Bernoulli distribution could be used to calculate the expected value of the number of attempts required.

$$E[n_a] \equiv \sum_{i=0}^{\infty} i (1-p)^{i-1} p = \frac{1}{p}$$

This is the same formula used in telecommunications analysis to calculate the expected number of retransmissions for a given error probability.

However, while we speak of the average number of trials for success, there may be a difference in the cost of successive trials. Even a trial that does not yield success may yield information that lowers the cost or improves the chances of success for later trials. For example, in an exhaustive attack against a password, each trial reduces the size of the remaining space and improves the probability of success of the next trial.

In addition, protective measures may not operate the same on all trials. For example, modern systems include a number of measures that are intended to raise the cost of successive trials. These include:

- delay the prompt after a failed trial, reducing the effective bandwidth and increasing the cost of successive trials⁴.
- set an alert threshold at a specific number of failed logon attempts for a given userid and then revoking that userid; and
- break the connection after a threshold number of failed attempts, thus adding the cost and delay of a re-connection to the next trial.

Note that these techniques are not intended as an impregnable defense. Rather they are intended to raise the cost of attack. However, in a target-rich population they succeed in moving the attacker to other targets with lower cost of attack.

The model must also account for special knowledge or experience available to the attacker. For example, attackers know that some passwords are far more likely than others. In the population attacked by the Wank worm, one system in five had at least one password equal to the null password or to the user ID. Short dictionary (or "sweet list") attacks are a staple⁵. In order to extend the model to reflect the variable costs, we first consider the cost of the attack, which is equal to the sum of the individual trials, over the expected number of attacks:

$$c_a = \sum_{i=0}^{E[n_a]} C_i$$

To calculate each C_i value, we need to consider how the above cost increasing techniques can be modelled. As a simplifying assumption we will consider that the primary cost associated with an attack is driven by the time required for the attack. This is appropriate, for example, if the attacker had to pay for a long distance telephone call during the attack. It is also appropriate if the chance of detection is proportional to the duration of the attack.

A realistic cost formula can be formulated from the following variables:

$K_{\Delta t}$:=	linear cost factor
$MAAT$:=	Mean active attack time. This includes the number of characters required to send a userid/password pair to the abstract machine's user interface, divided by the network bandwidth.
$FailDelay$:=	Time delay inserted after a failed access.
$NumTries$:=	Number of tries allowed before the communications link is severed.
$ReconnectTime$:=	Delay cost associated with connecting/reconnecting to the machine.
$KillThreshold$:=	number of attempts allows before account is invalidated.
i	:=	Attack iteration number.

$$C_i = \begin{cases} K_{\Delta t} [(MAAT + FailDelay) + (\text{if } (i \bmod NumTries = 0) \text{ then } ReconnectTime \text{ else } 0)] & \text{if } i < KillThreshold \\ \infty & \text{if } i \geq KillThreshold \end{cases}$$

Clearly if the expected number of attacks, $E[n_a]$, is greater than the $KillThreshold$, the individual C_i terms become infinite, and the summation also becomes infinite. For those cases that are interesting, this yields:

$$E[C] = K_{\Delta t} \sum_{i=1}^{E[n_a]} (MAAT + FailDelay) + K_{\Delta t} \sum_{i=1}^{\frac{E[n_a]}{NumTries}} ReconnectTime$$

$$E[C] = E[n_a] K_{\Delta t} \left((MAAT + FailDelay) + \frac{ReconnectTime}{NumTries} \right)$$

THE VALUE OF SUCCESS

A successful attack gives the attacker access to whatever capacity, paths, abstractions and recorded data that are implemented in the target. For purposes of the model it is necessary to divide these resources into those which reduce the cost of attack against the network and those which do not. While the latter may be intrinsically valuable and influence the attackers propensity to attack this particular machine, only the former influences the security of the network.

Capacity

Capacity is the most fungible and easy to measure and appreciate of the values in the target. Until consumed, i.e., used or wasted, it can be devoted to any purpose. For example, it can be used to reduce the cost of attack against another system.

Identity of the Victim

The successful attacker obtains the identity of the victim node. That is, he gains the ability to appear as that node to other nodes in the network. While under the compromised identity, the attacker is accorded all the trust and tolerance that would be accorded to the victim.

At a minimum, this is the tolerance that would be accorded to any member of the network or community. Historically the network has been both orderly and trusting. Most of the behavior has been orderly. In part because of this and in part because the network was new and small, users have been tolerant of eccentric behavior on the part of other users. They have also been accomodating and helpful to other members of the community. Therefore, simply being able to act under the identity of a legitimate member of the community is helpful to an attacker.

At the most, the attacker may gain all of the privileges and trust accorded to the most privileged and trusted administrator.

Identity of the Information

The amount of information that the attacker gets from coded data is, in part, a function of what he already knows. While a bit is the amount of information that reduces uncertainty by half, the recorded code of that bit is valuable only if one is able to recognize, i.e., identify, the bit.

Currency of the Information

The value of most data will go down with age. For example, even reusable passwords may have a finite life. The later in its life it is learned, the lower its value. This leads to at least three alternative models for information currency:

- Linear decay where the value of the information decays at a constant (and potentially zero) rate;
- Exponential decay, perhaps using a half-life model or other exponential decay; and
- Abrupt expiration, where at a specific point, the value drops instantly to zero.

Combinations of two or more of these models may be suitable for specific real world problems. For example, a userid/password pair that will expire at a fixed point in the future provides a capacity value that linearly decreases. Exponential decay is suitable for information that loses relevance over time, such as a data store that contains the network topology as of a given date in the past. Since the network is continually revised, the information becomes outdated, the general information in the data store will remain valuable after the detailed information is obsolete.

Scope or Quantity

All other things being equal, the more data, or the wider its scope, the more value. The value of data will normally rise along an "S" shaped curve with quantity. Thus, the more other systems a given target knows about, the greater the value of a successful attack against it. The initial, small amounts of information do not yield much knowledge (or value). As a critical mass of information is gathered, each datum provides an incremental increase in value. When nearly all the information has been captured, the incremental benefit tapers off. Thus the more other systems a given target knows, the greater the value of a successful attack against it.

Development of a formula that maps a linear estimate of information scope onto the "S" curve is easy. The difficult part is development of a quantitative metric that maps a given amount of information to an estimate of the percentage of the abstract machine's total information.

MODELLING THE VALUE OF SUCCESSFUL ATTACK

In modelling the value of successful attack, we make three useful distinctions. First we distinguish that part of V that is useful in directly lowering the cost of attack against the network from any other resources. For example, information about the network or other machines will lower the cost of attack against those machines, while otherwise very valuable financial information will not. This latter data may have intrinsic value which adds to the attractiveness as a target of the machine, but does not impact the security of the network.

Second, we distinguish between that portion of V which can be modelled as another abstract machine and that which cannot. For example, if access to machine A includes cheaper access to B and C than is available without access to A , then we would model that access as abstract machines with the new, lower, cost of attack.

Third, we distinguish between those resources whose identity is likely to be known outside the abstract machine. For example, if an abstract machine is an instance of a population of similar machines, then identities which it shares with those other machines will be widely known and exploitable. For example, information in a TCP/IP "host table" would be readily identifiable and exploitable, as it was in the case of the Internet worm.

MODELLING A NETWORK

In this section we define a network, extend the model, and apply it to the network as defined. The definition of network that we suggest is special, but it does include the networks of interest.

DEFINITION OF A NETWORK

We define a network as two or more of our abstract machines connected in such a way that the users of one can see the user interface of the others, or an aggregation of such networks so connected. We deal with the collection of things connected, rather than with the connecting infrastructure. We deal with networks of computers, not networked computing.

For the moment and for our purposes, we define the network to be so physically secure, protected, or isolated that the only viable attack is via the user interfaces. By this means we exclude from the model the issues of eavesdropping on the media, violence, subornation, or coercion. We assert that these issues are general to remote computing of any kind, rather than peculiar to networking of computers. We recognize that there are some among our possible readers who believe these to be the most interesting issues. We beg their indulgence and patience. We suspect that by binding some other variables in the model constant, it can later be used to examine these interesting issues.

This definition would not treat a single, abstract machine as a network. It would distinguish between a network and a collection of such abstract machines based upon whether or not the machines were connected in such a way that the user of one machine could connect to and exploit the user interface of another. Based upon this distinction, most multi-user machines would not qualify as networks, since one process is not normally able to connect to another without its knowledge and cooperation. On the other hand, two connected single user systems might qualify. Two connected multi-user systems might qualify as a complex network.

For the moment, and as a working hypothesis, we assert that for the purposes of our model, the most significant difference between whether these machines are two boxes connected by wire or are compartments in a single box is one of bandwidth. That is, all other things being equal, the link between two compartments in the same box will support more attack trials per unit time than will a link between two separate boxes. but that no other difference is important to our purpose.

For purposes of this work, it is essential that this definition preserve the significant properties of our abstract machine and model or reflect the security properties of networks of real systems. Our model is subject to examination and criticism to ensure that it does so. Nonetheless, it is in the nature of a model to emphasize some properties of that being modelled over others.

DEFINITION OF NETWORK SECURITY

As with the single abstract machine, such a network is defined as being attack secure if the value of a successful attack is less than the cost of such an attack.

Note that such a level of security does not necessarily prevent all attacks. It does not make the network resistant to all attacks. It does not even prevent successful attacks against some systems in the network. On the other hand, neither does it require that all components of the system be attack secure in order to say that the network is secure.

The analogy here is to a fishnet rather than to a chain. While each is weakened by the compromise of a link, the net can continue to work in the face of many broken threads or knots. Once more, the purpose of the model is to be able to talk about relative, rather than absolute, security.

AN ILLUSTRATION

For purposes of clarification it may be useful to apply our model to some existing systems and attacks. Consider first one of the Unix systems that was attacked by the Internet worm³. Since those machines were not atomic machines, they were, for our purposes, either a collection of abstract machines, or a network of such machines. That is to say, *sendmail*, *fingerd*, *debug*, and each of the user processes would each be modelled as a separate abstract machine. They would not be modelled as a network because they were not connected in such a way that the user of one such process could see the user interface of another from that process. *Debug* would be modelled as an abstract machine within a machine, *sendmail*. On the other hand, any of these abstract machines would be modelled as a network with user processes in other Unix machines. The value of a successful attack against *debug* includes access to all of the processes within the Unix machine.

Note that the real cost of attack in this system was measured in terms of tens to low hundreds of man hours. This was obviously and certainly lower than the cost of the loss. Thus, the network was not protection secure. While no value was converted to the attacker, it is possible that a small variation on the attack might have yielded considerable resource. Therefore, by our definition the network was not attack secure. Note also that the penetration of one system lowered the cost of attack against subsequent ones. Once the worm succeeded in getting itself executed in a target machine, it gained a path to additional machines, knowledge about them, and capacity with which to attack them, i.e., it could and did start attack processes against them.

EFFECT OF NETWORKING ON SECURITY

We assert that, all other things remaining equal, the relative security of a system goes down when it connects to any other system. That is, the population of attackers with connection goes up, the cost of attack goes down, and the value of success goes up. The cost of attack goes down because there is more capacity to be used in the attack. The value of success goes up because the value included in the target goes up by the value of the connection to other targets.

SPATIAL RELATIONSHIP BETWEEN ABSTRACT MACHINES ON THE NETWORK

We assert that within the context of this model of abstract machines, the proper model is a fishnet, not the more common model of a chain. Specifically, a fishnet may have large holes, and may even have a number of large holes, while still providing its design function. With a fishnet, some holes cause problems, and others do not. This model of network security shares this characteristic.

As in a fishnet, the potential harm caused by a specific hole is related to the proximity of this weakness to others. Additionally, we suggest that the value of an attack is decreased with increasing distance. Many mathematical models are available that can show this decrease in value. We suggest that the inverse square law, that is often encountered in physical phenomena such as light or other electromagnetic emissions, is a suitable and well understood description for the decreasing value of an attack over distance. We propose that rather than a traditional geometric distance metric, an alternative metrics for distance are either "time" or "work." These terms are interdependent; either may be used to suit the specifics of the model under study. The amount of effort required to set up for an attack, or the exposure and thus the probability of detection during an attack, measures the cost to the attacker, and is used as the distance metric.

USES OF THE MODEL

We think that the ideas and abstractions presented in this paper have a number of useful applications. First, we submit that the simple abstract machine and simple security model provide a useful way to think about security, i.e., relative rather than absolute. Second, modeling a network as a collection of simple abstract machines enables us to isolate and examine properties of the network which are important to its security. Finally, they can be used to support the additional research which we recommend below.

SUGGESTED RESEARCH

This paper is entitled "Toward a Model..." That is intended to suggest that we do not think or suggest that this paper represents completed work. We suggest three paths.

First, as with most such models, this one can be elaborated in both detail and scope. For example, more components of the cost of attack can be identified and described. More important, the results of success can be identified, classified, evaluated, and described. The object would be to determine the resources that are of use to an attacker, how he might use them, how much they will reduce his cost of attack, and how that value to him may be limited or reduced. All of this needs to be rigorously and completely described mathematically.

Second, a simulation can be constructed based upon the model. We believe that the a network simulator can be built based upon the properties of the network components and the definition of security that we have described. This simulator would enable us to assess how the security of the network responds across time to attack.

Third, experiments can be conducted using the simulator. For example, a network of given topology, cost of attack, and value of success can be simulated. The simulator can be used to evaluate the resistance of the network to varying kinds of attacks or varying resources available to the attacker. By holding the form and resource of the attack constant, and varying the topology of the network, the experimenter will be

able to compare the resistance to attack of different topologies. By varying the Logon mechanisms of various nodes, the evaluator will be able to measure the cost to the attacker that results from different kinds of security mechanisms and determine the effect of their distribution within the net. That is, one could assess whether the resistance of the network to attack can be raised by raising the cost of attacking only some of the nodes. By holding everything else constant, and varying the privileges and information about other nodes in the abstract machines the simulator can be used to measure the effect of such measures as compartmenting privileges or encrypting password tables.

RECOMMENDED PRACTICES

This analysis suggests a number of practices that can improve the security of networks of computers. While raising the cost of attack these measures will have a minimum impact on legitimate users acting in the intended way. These practices include:

- Limit the life passwords; prefer one-time passwords for most uses in networks.
- Consider dual passwords for privileged accounts.
- Require cooperation of two people to alter logon programs.
- Automatically reduce the bandwidth in the face of possible attacks (delaying the prompt is the preferred mechanism because it corrects itself automatically; alternatively break the connection, revoke the user ID, or otherwise disable the logon);
- Do not offer any services or information to unknown or untrusted systems or users. Offer GUEST and ANONYMOUS services only from systems specifically intended for that purpose.
- Limit the information that is given to unauthenticated users. Specifically, do not grant access to logon IDs, do not tell the user what is wrong with a logon attempt, and do not tell them that an attack is suspected. Do not tell them the identity or type of the system or network.
- For security, compartment the network into subnets, name spaces, and domains that each require their own authentication.
- Prefer multiple user names and authenticators for sensitive applications; use single name spaces for user convenience.

REFERENCES

[1] Murray, William H., *On Computer Security and Public Trust*, Proceedings of the National Conference on Computing and Values; August 1992, New Haven, CT.

[2] Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria, National Computer Security Center, Ft. Meade, MD, 1987

[3] Spafford, Eugene H., *The Internet Worm Program: An Analysis*; ACM Computer Communication Review; January 1989; Number 19(1).

[4] Capek, Peter, *Discouraging Penetration Attempts on Interactive Computer Systems Without Denial-of-Service*, IBM Technical Disclosure Bulletin, February 1989, Volume 31, Number 9, Page 147-149.

[5] Stoll, Clifford, *The Cuckoo's Egg*, Doubleday 1990, New York, NY.