A preamble into aligning Systems engineering and Information security risk measures

Craig S Wright

School of Computing and Mathematics
Charles Sturt University
Wagga Wagga, NSW, Australia
crwright@csu.edu.au,

Abstract. For many years information security and risk management has been an art rather than a science. This has resulted in the reliance on experts whose methodologies and results can vary widely and which have led to the growth of fear, uncertainty and doubt within the community. At the same time, the failure to be able to effectively expend resources in securing systems has created a misalignment of controls and a waste of scare resources with alternative uses. This paper aims to introduce a number of models and methods that are common in many other areas of systems engineering, but which are only just starting to be used in the determination of information systems risk. This paper introduces the idea of using neural networks of hazard data to reliably model and train risk systems.

Keywords: Modeling, Hazard, non-homogeneous Poisson process failure intensity modeling.

1 Introduction

For many years information security and risk management has been an art rather than a science. This has been detrimental to the economy as a whole as well as to the operations of many organizations. The result has been a reliance on experts whose methodologies and results can vary widely and which have led to the growth of fear, uncertainty and doubt within the community. Although many true experts do exist who exhibit an insightful vision and ability, for each true expert, many inexperienced technicians and auditors abound.

This failure to be able to effectively expend resources in securing systems has created a misalignment of controls and a waste of scare resources with alternative uses. This paper aims to introduce a number of models and methods that are common in many other areas of systems engineering, but which are only just starting to be used in the determination of information systems risk. These processes can help both the inexperienced security professional as well as adding to the arsenal of tools available to the consummate expert.

Where possible, the standard systems reliability engineering terms have been used in this paper. These formula and methods have been widely used in systems

engineering, medicine and numerous other scientific fields for many years. The introduction of these methods into common use within risk and systems audit will allow the creation of more scientific processes that are repeatable and do not rely on the same individual for the delivery of the same results.

2 System Survival

When assessing network reliability, it is necessary to model the various access paths and survival times for not only each system, but for each path to the system. This requires the calculation of the following quantitative fields

- R(t) The Reliability function
- MTBF Mean Time Between Failures
- MTTF Mean Time to Repair/Fix
- λ The expected survival rate [24]

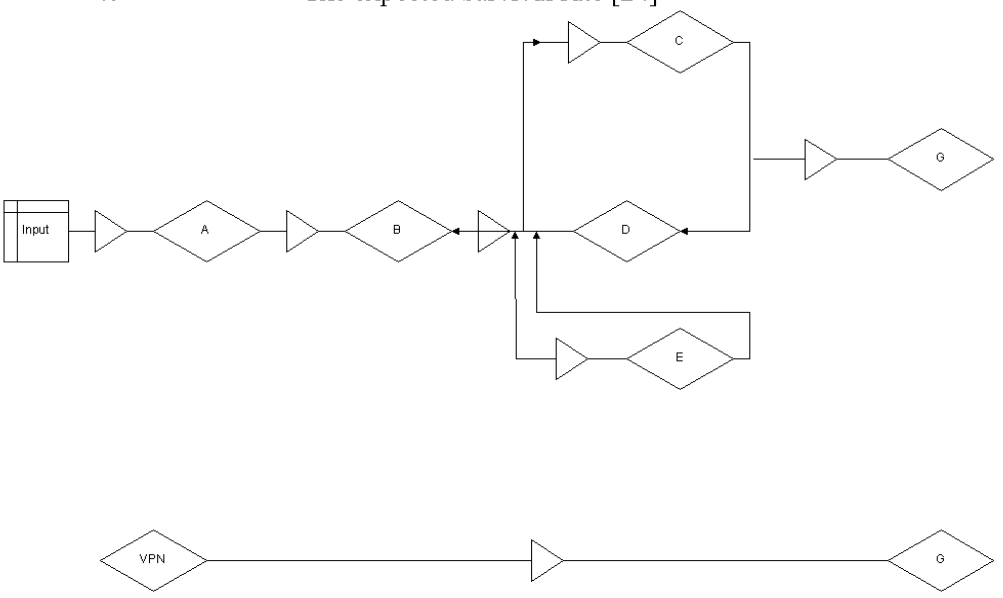

Fig. 2.1. In this example we model attacking a series of systems. In this example we have two separate paths into the critical data. An attacker can either use the VPN and bypass the firewall and other defensive controls; or the attacker can chain an attack through multiple systems.

Other measures will be introduced later. The expected survival or failure rate λ is used throughout this paper and is detailed further in EQ 3.6 and EQ 3.7. Where possible, the standard systems reliability engineering terms have been used. In the case of a measure such as the MTTF, this represents the time both to discover and recover a compromised system. The true value estimate for the system comes as a measure of the applications on the system, this may be estimated for a less economically expensive (though less accurate) estimate. In this calculation, the compromise measure, MTBF is best thought of as the mean time to the first failure.

This can be modelled with redundancy in the design. Here, each system is a parallel addition to the model. Where a system is required to pass another, a serial measure is added. For instance, if an attacker has to:

- bypass system A (the firewall) to
- compromise system B (an authentication server) which allows
- an attack against a number of DMZ servers (C, D and E) where
- systems C and D are connected to the database through
- a secondary firewall (system F) to (Not in figure 2.1)
- the database server G (as displayed in figure 2.1).

The attacker can either attack the system directly through the VPN or by following the attack path indicated by the systems. If the firewall system A restricted the attacker to a number of IP addresses, the attacker may do 1 of a number of things in attacking this system (in order to gain access as if the attacker was one of these IPs):

1. Compromise the input host
2. Spoof an address between the input IP address (such as through a router compromise at an ISP or other system)
3. Compromise the VPN

Other options, such as spoofing an address without acting as a MITM (Man In The Middle) will leave some attacks possible that can not result in a compromise of system G. These could have an economic impact that would be calculated separately. Such an event that can be calculated would be a DDoS (Distributed Denial of Services) attack on the server.

Hence, the effective attack paths are:

- Input, A, B, C, F, G
- Input, A, B,D, F, G
- Input, A, B, E, C, F, G
- Input, A, B, E, D, F, G
- VPN, G

In this instance, it is necessary to calculate conditional probabilities as these paths are not independent. Here the options need to consider first include (the paper will use the term I to define an attack on the Input system and S to refer to a spoofed attack of the input system):

- The conditional probability [1] of compromising system A given a successful spoof attack on the Input system, $P(I \cup A_1) = P(I).P(A_1 \mid I)$ (where A_1 refers to an attack on system A using path No. 1, or Input, A, B, C, F, G)
- The conditional probability of attacking system A
- The probability of attacking system G, $P(V \cup G_5) = P(V).P(G_5 \mid V)$

Each of the attack paths are able to be treated as independent. Hence, the overall probability of an attack is a sum of the conditional probabilities from each attack path. As a consequence, the attacker will most likely come over the lowest cost path, but the probabilistic result allows for an attack from any path. The high and low probability attack measures are jointly incorporated in the process.

Presuming no other paths (such as internal attacks etc) it is feasible to model the alternate probability as not possible (or at least feasible). Here $P_6 = 0 = \in$. Additionally, the probability of an attack over path 5 (the VPN) can be readily calculated without further input as:

$$P_5 = P(V \cap G_5) = P(V).P(G_5 \mid V)$$

Here:

$$P(V) = e^{-\lambda_V t} + \left(-\lambda_V t\right) e^{-\lambda_V t}$$

[3] and

$$P(G_5) = e^{-\lambda_G t} + \left(-\lambda_G t\right) e^{-\lambda_G t}$$

$$P(G_5 \mid V) = \prod P(G_5)$$

EQ 2.0

Here there exists a single system behind the VPN. Where more than one system exists, it is necessary to calculate the joint probability as is detailed below. In the example, with only a single system:

$$P(G_5 \mid V) = \prod P(G_5) = P(G_5) \times 1$$

$$\therefore P(G_5 \mid V) \rightarrow P(G_5)$$

EQ 2.1

Equation 2.1 holds as the probability of the attacker compromising system G when the VPN has been compromised approaches 1. This is as the attacker has a single target with the VPN and the utility of attacking the VPN and no more is negated as no other systems exist and the VPN offers no other utility for the attacker alone.

The values, λ_V and λ_G are the expected survival time or the mean time to compromise for the VPN and database respectively as configured and *t* is the amount of time that has passed from install and represents the current survival time of the system [16].

Here:

$$P_5 = P(V \cap G_5) = P(V).P(G_5 \mid V)$$

$$P(G_5 \mid V) =$$

$$= e^{-(\lambda_G + \lambda_V)t} - \left(\lambda_G + \lambda_V\right) t e^{-(\lambda_G + \lambda_V)t}$$

EQ 2.2

On the other hand, the probability of compromise to system I is based on the number of systems and as $n_I \rightarrow \infty, P(I) \rightarrow 1$. Basically, as more systems are allowed to connect to system A, the closer the probability of compromise tends towards a value of P=1. That is, as the systems available to be compromised increase, the probability of compromise approaches certainty. This is generalised as each addition to the system adds a positive probability of compromise when added to an existing system [3].This occurs as no system can be shown to have a probability of compromise P=0. Hence, for each additional component added into the system, the chance of compromise approaches P=1 (where it is finally reached at n=∞).

Where there are only a limited number of systems, the probability can be computed as a sum of the systems. Where there are a large number of systems with equivalent (or at least similar) properties, these can be calculated through the sum of the systems.

If in the above example, system E is replaced with a series of systems (each with the same configuration) it is possible to calculate the probability of a compromise of one of the "E" systems as follows:

$$P(E) = R(E) = 1 - \prod_{i=1}^{n} \left[1 - P\left(E_i\right) \right]$$

EQ 2.3

Here, P(E) is a multiplicative and not additive function. As such, if system "E" is defined as a DNS server with a single BIND service and SSH for management of the host, an attacker has two means to compromising the system;

- Attack SSH
- Attack BIND

The probability can be considered as independent in this case if there are no restrictions. In the example, DNS is an open service, that is, P(I)=1. The SSH service may or may not be open and could be restricted. If this is the case 0<P(I)<1. In the simple case where no restrictions have been imposed on SSH, the probability can be calculated as a standard independent probability formula. This is:

$$P(SSH) = e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}$$

$$P(DNS) = e^{-\lambda_{DNS}t} + \left(-\lambda_{DNS}t\right)e^{-\lambda_{DNS}t}$$

$$P(E) = 1 - P(SSH).P(DNS)$$

$$\therefore P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} + e^{-\lambda_{SSH}t}\left(-\lambda_{DNS}t\right)e^{-\lambda_{DNS}t} + e^{-\lambda_{DNS}t}\left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t} \right]$$

$$P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS}t\right)e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{SSh}t\right)e^{-(\lambda_{SSH}+\lambda_{DNS})t} \right]$$

$$P(E) = 1 - \left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS} - \lambda_{SSh}\right)te^{-(\lambda_{SSH}+\lambda_{DNS})t} \right]$$

EQ 2.4

The complication comes where one of the services has been restricted as a further control. This is a combination of the probability of compromising the restrictions on the service (that is spoofing or otherwise bypassing IP address controls) and the compromise of the service itself. This can be represented by:

$$P(SSH) = \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t} \right].P(I)$$

$$\begin{aligned} \therefore P(SSH) &= \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t} \right].\left(1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right) \\ &= \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t} \right] - \left(\prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right).\left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t} \right] \end{aligned}$$

In this case, there exists a probability $P(I) = 1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]$ where the allowed source systems (I) are limited to a total of "n" IP addresses (or keys). The probability $P\left(I_i\right)$ of any source system being compromised will vary, but may be estimated based on the type and location of each system. As more systems are added

into the equation, the polynomial equation becomes more complex. In the event that similar systems are also accessing this, these can be calculated and the equation simplified.

For example, if two (2) classes of systems exist (Linux and Windows Vista) that comprise the set of systems I_i for a total of 4 systems (2x Windows and 2x Linux) these can be defined using:

$$P(I_1) \simeq P(I_2) = e^{-\lambda_{Win}t} + \left(-\lambda_{Win}t\right)e^{-\lambda_{Win}t}$$

&

$$P(I_3) \simeq P(I_4) = e^{-\lambda_{Linux}t} + \left(-\lambda_{Linux}t\right)e^{-\lambda_{Linux}t}$$

In the case where $P(I_1) \simeq P(I_2) = 0.25$ and $P(I_3) \simeq P(I_4) = 0.2$ due to the system configurations and patch status, it is possible to calculate P(I):

$$P(I) = \left(1 - \prod_{i=1}^{n}\left[1 - P\left(I_i\right)\right]\right)$$

$$= 1 - \left[(1-0.25)^2.(1-0.2)^2\right]$$

$$= 1 - \left[(0.75)^2.(0.8)^2\right] = 1 - \left[0.5625 \times 0.64\right] = 1 - 0.36$$

$$= 0.64$$

EQ 2.5

In this case, the probability of a compromise due to SSH would become:

$$P(SSH) = \left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right].P(I)$$

$$= 0.64\left[e^{-\lambda_{SSH}t} + \left(-\lambda_{SSh}t\right)e^{-\lambda_{SSH}t}\right]$$

EQ 2.6

With the details from the example at 2.2, it is possible to calculate the survival function for system E:

$$P(E) = 1 - 0.64\left[P(SSH).P(DNS)\right]$$

$$\therefore P(E) = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + \left(\lambda_{DNS}\lambda_{SSH}\right)t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - \left(\lambda_{DNS} - \lambda_{SSh}\right)te^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$

EQ 2.7

Thus, there exists a method to calculate the probability of each system as well as the conditional probability of that system.

The addition of a device (such as an IDS) changes or otherwise impacts *t* and adds additional complexity to the calculations. An IDS for instance can limit the value of *t* through a probabilistic feedback process. The more effective the IDS is, the quicker an attack or other incident will be intercepted. In this instance, *t* becomes a probabilistic function based on how effective the IDS itself is. This becomes a combination of the following factors:

- The inherent accuracy of the IDS (which is a trade-off between TYPE I and TYPE II errors [22] and it is a cost function in itself)
- The missed detection rate (even where an incident is noted, the analyst may miss the detection. As more false negatives are seen, the missed detection

rate increases (Ikeda, & Watanabe, 1962). As a result, increasing false negatives to capture all possible attacks ends in a limit where the IDS is no longer effective).

A Type I error is often denoted as a "*false positive*". This involves incorrectly rejecting the null hypothesis in favor of the alternative. Where an IDS is involved, a false positive would involve detecting and alerting on an event that did not actually happen to be an incident or attack. A Type II error is the opposite of a Type I error. A Type II error in an IDS system involves the false acceptance of the null hypothesis and is commonly referred to as a "*false negative*". It would imply that the packet or traffic is not an attack and is safe when it is in fact malicious or otherwise dangerous.

The IDS forms a cost function as the increase in reporting results in a greater number of false positives that need to be investigated. In limiting the false positives, the likelihood of missing an incident of note also increases. Each validation of a false positive takes time and requires interaction from an analyst. Hence the tuning of an IDS is balanced on maximizing the detection against cost.

In the event that the IDS does not detect the attack, the function mirrors that of the system without the IDS in effectiveness. Note that the cost of the system with the IDS is greater than the system without IDS. As a result, the addition of IDS is a limiting function. An increase in cost adds to the power of the IDS. This is, more analyst time and more detection capability lowers the false negative and false positive rate through an increase in cost. Each IDS system has an expected TYPE I and TYPE II error rate that will vary as the system is tuned to a particular environment. The result of this is an individualistic function for the organisation that can only be generally approximated for other organisation (even when the same IDS product is deployed).

For a given probability of survival, it is possible to calculate the expected survival time (t) of the system. This process becomes computationally infeasible in large systems with numerous inputs. For instance, on system E (as defined in EQ 2.3) it is feasible to rearrange the equation of the expected probability of system E being compromised. For instance, if a calculation of the expected function of survival time for a set survival probability P is desired, rearrange the equations in EQ 2.3 as follows.

$$P(E) = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$

$$\therefore if\ P = 0.99, \quad 0.99 = 1 - 0.64\left[e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t}\right]$$

$$or \quad e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t} = 0.0015625$$

$$\ln(e^{-(\lambda_{SSH}+\lambda_{DNS})t} + (\lambda_{DNS}\lambda_{SSH})t^2 e^{-(\lambda_{SSH}+\lambda_{DNS})t} - (\lambda_{DNS} - \lambda_{SSh})te^{-(\lambda_{SSH}+\lambda_{DNS})t}) = -6.4615$$

EQ 2.8

This result is in the form of:

$$At + B\ln(t) = C$$

From EQ 2.8 it is clearly seen that as $t \to \infty$ $[t + \ln(t)] \xrightarrow{t\to\infty} t$. From these equations, as long as t is large, an approximation can be deployed to obtain a lower

limit estimate of $At + B\ln(t) = C$ as $At \simeq C$. As such an approximate for the lower limit of time for system E's survival is defined as:

$$t = \frac{6.4615 + \ln\left(\lambda_{DNS} + \lambda_{SSH}\right)}{2\left(\lambda_{SSH} + \lambda_{DNS}\right)}$$

EQ 2.9

In EQ 2.4, it is demonstrated that the lower the value of t, the greater the error. Measuring "*t*" in seconds and substituting normal system values of λ allows for the use of Monte Carlo simulations to approximate the expected value of *t*.

For simplicity, let R represent reliability and Q the unreliability (hence, 1-R=Q).

For each application, a possibility exists to use Bayes' theorem to model the number of vulnerabilities and the associated risk. A mathematical introduction to Bayes' Theorem is available online from Weisstein [26] and in more detail from Joyce [17]. For open ports, the person evaluating risk can use the expected reliability of the software together with the expected risk of each individual vulnerability to model the expected risk of the application. For instance, it is conceivable to model $P(SSH)$ using this method.

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

EQ 2.10

alternatively;

$$P(A \cap B) = P(B)P(A \mid B) = P(A)P(B \mid A)$$

Over time, as vulnerabilities are uncovered and fixed (assuming that new vulnerabilities have not been introduced), fewer issues will remain. Hence, the confidence in the software product increases. This also means that mathematical observations can be used to produce better estimates of the number of software vulnerabilities as more are uncovered.

It is thus possible to observe the time that elapses [11] since the last discovery of a vulnerability. This value is dependent upon the number of vulnerabilities in the system and the number of users of the software. The more vulnerabilities, the faster the discovery rate of bugs. Likewise, the more users of the software, the faster the existing vulnerabilities are found (through both formal and adverse discovery).

2.1 Mapping Vulnerabilities within software

Now let *E* stand for the event where a vulnerability is discovered within the Times *T* and *T*+*h* for n vulnerabilities in the software

$$P(E \mid n) = \int_{T}^{T+h} n\alpha e^{-n\alpha t}\, dt \approx n\alpha e^{-n\alpha T} h$$

Where a vulnerability is discovered between time *T* and *T*+*h* use Bayes' Theorem to compute the probability that *n* bugs exist in the software:

$$P(n_{vulnerabilities} \mid E) = \frac{ne^{-(n\alpha T+\beta)}\frac{\beta^n}{n!}}{\sum_{n=0}^{\infty}\left[ne^{-(n\alpha T+\beta)}\frac{\beta^n}{n!}\right]}$$

EQ 2.11

From this it can be seen that:

$$P(n_{vulnerabilities} \mid E) = \frac{\frac{\left(\beta e^{-\alpha T}\right)^{n-1}}{(n-1)!}}{\sum_{n=0}^{\infty}\left[\frac{\left(\beta e^{-\alpha T}\right)^{n-1}}{(n-1)!}\right]}$$

EQ 2.12

EQ 2.12 will apply for all versions of software [29]. As patches and updates are applied to the software, existing vulnerabilities will be rectified and removed, but new flaws related to how many new lines of code have been added in the patching process will be introduced and will also need to be calculated.

By summing the denominator it can understood that in observing a vulnerability at time T after the release and the decay constant for defect discovery is α, then the conditional distribution for the number of defects remaining is a Poisson distribution with expected number of defects $\beta e^{-\alpha T}$.

Hence:

$$P_{\beta e^{-\alpha T}}(n) = e^{\beta e^{-\alpha T}}\frac{\left(\beta e^{-\alpha T}\right)^n}{n!}$$

EQ 2.13

This can be extended to create a method to calculate the expected failure of a system based on the interaction of multiple software products.

3 Exponential Failure

The reliability function (also called the survival function) represents the probability that a system will survive a specified time *t*. Reliability is expressed as either MTBF (Mean time between failures) or MTTF (Mean time to failure). The choice of terms is related to the system being analysed. In the case of system security, it relates to the time that the system can be expected to survive when exposed to attack. This function is hence defined as:

$$R(t) = 1-F(t) \qquad \text{EQ 3.1}$$

The function F(t) in EQ 3.1 is the probability that the system will fail within the time 't'. As such, this function is the failure distribution function (also called the unreliability function). The randomly distributed expected life of the system *'t'* can be represented by a density function, $f(t)$ and thus the reliability function R(t) can be expressed as:

$$R(t) = 1 - F(t) = \int_t^{\infty} f(t)dt$$

EQ 3.2

The time to failure of a system under attack can be expressed as an exponential density function:

$$f(t) = \frac{e^{-t/\theta}}{\theta}$$

EQ 3.3

where θ is the mean survival time of the system when in the hostile environment and t is the time of interest (the time that the user wishes to evaluate the survival of the system over). Together, the reliability function, R(t) can be expressed as:

$$R(t) = \int_t^{\infty} \frac{e^{-t/\theta}}{\theta} dt = e^{-t/\theta}$$

EQ 3.4

The mean (θ) or expected life of the system under hostile conditions can hence be expressed as:

$$R(t) = \int_t^{\infty} e^{-t/M} dt = e^{-t\lambda}$$

EQ 3.5

Where M is the MTBF of the system or component under test and λ is the instantaneous failure rate [4] where Mean life and failure rate are related by the formula:

$$\lambda = \frac{1}{\theta}$$

EQ 3.6

The failure rate for a specific time interval can also be expressed as:

$$\lambda = \frac{\#Failures}{\sum Operating \quad Hours}$$

EQ 3.7

Failure rates are generally expressed in terms of failures per hour, percentage of failures per each 1,000 hours or the rate of failures per million hours. For instance, if a system has a 90 day patch cycle (the total mission time) and that the total number of software failures in that time is expected to be (or is later measured to be) 6 vulnerabilities, it is conceivable to calculate the failure rate per hour as:

$$\lambda = \frac{6}{90 \times 24} = \frac{6}{2,160} = 0.002778$$

EQ 3.8

In the case of an exponential distribution for the system mean survival under attack, the MTBF can be defined as:

$$MTBF = \frac{1}{\lambda} = \frac{1}{0.002778} = 360 hours$$

EQ 3.9

Hence, it is expected the system to survive 15 days before a vulnerability is discovered. This does not return when a system will actually be exploited, simply the expected probabilistic time that can be used to project and plan future expenditure.

4 Automating the process

The main advantage to a systems engineering approach is the ease with which it can be automated. The various inputs and formula noted throughout this paper can become inputs into a neural network algorithm (Fig. 4.1). Equation (2.1) could be modeled in three layers (Fig 4.2).

Here, an input layer with one neuron for each Input (system or application) could be used to map for IP Options, Malware and Buffer overflow conditions, selected attacks etc. The system of perceptrons would be processed using a hidden neuron layer in which each neuron represents combinations of inputs and calculates a response based on current data coupled with expected future data, a prior data and external systems data. Data processed at this level would feed into an output layer. The result of the neural network would supply the output as an economic risk function.

In this way, a risk function can be created that not only calculates data based on existing and known variables [12], but also updates automatically using external sources and trends. Many external sources (http://www.dshield.org) have become available in recent years that provide external trending and correlation points. Unfortunately, most of these services have clipped data as the determination of an attack is generally unclear and takes time to diagnose where much otherwise useful data is lost. When monitoring the operation of a system or the actions of uses, thresholds are characteristically defined above or below which alerting, alarms, and exceptions are not reported. This range of activity is regarded as baseline or routine activity.

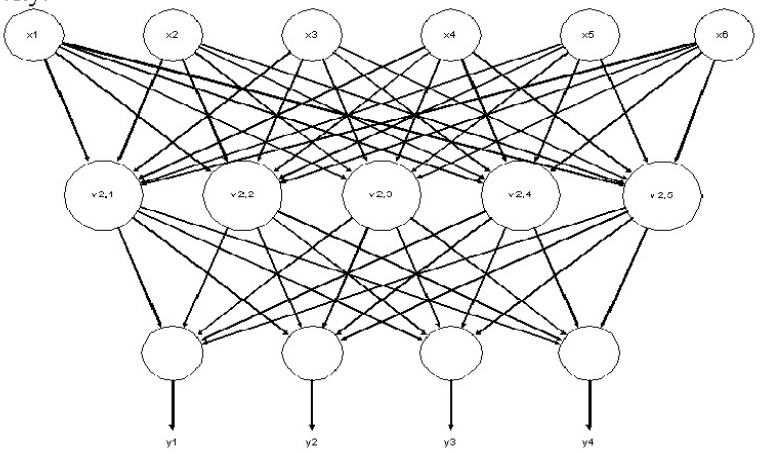

Fig 4.1 A depiction of a Multi-Layer layer topology neural network

Multi-Layer layer topology neural networks can be used to accept data from risk models and automatically update the risk profile of an organization. In modeling risk, each application and system can be modeled using a perceptron.

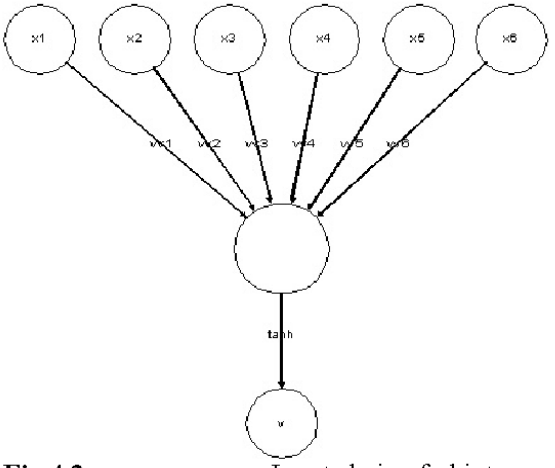

Fig 4.2 Inputs being fed into a perceptron.

The perceptron is the computational workhorse in this system. In this it is reasonable to model the selected risk factors for the system and calculate a base risk that is trained and updated over time. The data from multiple organizations can be fed into a central system [18] that can be distributed to all users. This could be integrated and sold as a product enhancement by existing vendors or independent third parties could maintain external datasets.

$$v_{i,j} = f(\sum_{k=0}^{n} w_{i,j,k} \cdot x_k \ )$$

EQ 4.1

EQ 4.1 defines the input variables as,

- $x_1 \ldots x_n$ are the inputs of the neuron,
- $w_{i,j,0} \ldots w_{i,j,n}$ are the weights,
- f is a non-linear activation function,
- hyperbolic tangent (tanh),
- $v_{i,j}$ is the output of the neuron.

A large vendor such as Microsoft could create an implementation model. In place of offering stale recommended security settings (such as currently occurs with Microsoft's MBSA), the risk application could automatically collect data from user systems on patch levels and group policy configurations and utilize these in order to calculate and report on an estimated level of risk and an expected survival time for the system in a number of different scenarios. For instance a notebook computer could have a set of risks. This would include the risk when connected to the corporate network, when connected to a wireless hotspot etc.

The training of the network would require the determination of the correct weights for each neuron. This is possible in selected systems, but a far larger effort would be required to enable this process for more generalized deployment. The data needed for such an effort already exists in projects such as DShield, the Honeynet Project and in many similar endeavors. The question is whether there truly exists a will as a community to move from an art to a science.

5 Conclusion

The equations presented in this paper allow organizations to compare the deployed risk strategies against both their own historical data and that of third parties. In this manner, strategy can be formulated in order to optimize audits and system reviews in a manner that detects an incident in the most economical manner. Projects are all risk derived exercises and if our profession can better manage and calculate risk, society will benefit.

Modeling the failure rate of systems and the propagation rate of an attack, allows us to calculate an expected number of hosts that are anticipated to have been compromised in the time between an audit given a specified survival function or threat. Past data and comparisons from similar systems (such as survival data from http://www.dshield.org/reports.html) allow for the modeling of alternative systems where a reported number of events have been reported against those deployed.

Dependence, variation, randomness, and frailty add to the risk toolset of multivariate failure event analysis. Using frailty theory to model information system risk allows us to better predict risk and to more effectively allocate scarce resources through selecting the most economically viable targets to defend as well as choosing the optimal detection strategies. The properties of censoring-handling and frailty modeling have turned multivariate survival analysis into an exceptional tool for the determination of system risk.

For decades, information security practitioners have engaged in qualitatively derived risk practices due to the lack of a scientifically valid quantitative risk model. This has led to both a misallocation of valuable resources with alternative uses and a corresponding decrease in the levels of protection for many systems. Using a combination of modern scientific approaches and the advanced data mining techniques that are now available provides the technologies and data to create a new approach to information systems risk and security.

The optimal distribution of economic resources across information system risk allocations can only lead to a combination of more secure systems for a lower overall cost. The reality is that, like all safety as an issue, information security is based on a set of competing trade-offs between economic constraints. The goals of any economically based quantitative process are to minimize cost and risk through the appropriate allocation of capital expenditure. To do this, the correct assignment of economic and legal liability to the parties best able to manage the risk (this is the lowest cost insurer) is essential and needs to be assessed. This will allow insurance firms to develop expert systems that can calculate risk management figures that can be associated with information risk. This will allow for the correct attribution of information security insurance products that can be provided businesses generally.

Externality or the quantitative and qualitative effects on parties that are affected by, but who are not directly involved in a transaction is likewise seldom quantified, but is an integral component of any risk strategy. The costs (negative) or benefits (positive) that apply to third parties are an oft overlooked feature of economics and risk calculations. For instance, network externality (a positive effect that can be related to Metcalfe's law; value of a network = 2 times the network's number of users) attributes positive costs to most organizations with little associated costs to themselves. In these calculations, the time-to-market and first-mover advantages are

critical components of the overall economic function with security playing both positive and negative roles at all stages of the process.

The processes that can enable the creation and release of actuarially sound threat risk models that incorporate heterogeneous tendencies in variance across multidimensional determinants while maintaining parsimony already exist in rudimentary form. Extending these though a combination of Heteroscedastic predictors (GARCH/ARIMA etc) coupled with non-parametric survival models will make these tools more robust. Effort needs to be expended in the creation of models where the underlying hazard rate (rather than survival time) is a function of the independent variables (covariates). Cox's Proportional Hazard Model with Time-Dependent Covariates would be a starting point, with a number of non-parametric methods available where cost allows.

As we move further into the 21st century, it is time we as a profession started to model risk as a scientific process and move away from the art based cottage industry that exists right now. This paper has presented a number of methods that can be used to gauge the expected failure events in a system of computer hosts.

References

1. Annis, C. (2010) "Joint, Marginal, and Conditional Distributions" Retrieved October 11, 2011, from http://www.statisticalengineering.com/joint_marginal_conditional.htm
2. Benveniste, A. and Jacod, J. (1973). Systèmes de Lévy des processus de Markov. *Invent. Math.* **21** 183--198. Mathematical Reviews
3. Blanchard, B. and Fabrycky, W. (2006) "Systems Engineering and Analysis", 4th Ed. Prentice Hall International Series in Industrial and Systems Engineering, USA
4. Brémaud, P. (1981). *Point Processes and Queues: Martingale Dynamics*. Springer, New York. Mathematical Reviews
5. Chakrabarty, A. and Guo, X. (2007). A note on optimal stopping times with filtration expansion. Preprint, Univ. California, Berkeley.
6. Corcuera, J. M., Imkeller, P., Kohatsu-Higa, A. and Nualart, D. (2004). Additional utility of insiders with imperfect dynamic information. *Finance and Stochastics* **8** 437--450.
7. Dellacherie, C. and Meyer, P. A. (1982). *Probabilities and Potential. B*. North-Holland, Amsterdam.
8. Dempster, A.P. and Laird, N.M. and Rubin, D.B. (1977).Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, B39.
9. Elliott, R. J., Jeanblanc, M. and Yor, M. (2000). On models of default risk. *Math. Finance* **10** 179--195.
10. Grandell, J. (1991) "Aspects of Risk Theory". Springer, New York
11. Guo, X., Jarrow, R. and Zeng, Y. (2005). Modeling the recovery rate in a reduced form model. Preprint, Cornell Univ.
12. He, S. W., Wang, J. G. and Yan, J. A. (1992). *Semimartingale Theory and Stochastic Calculus*. Science Press, Beijing.
13. Ikeda, N. and Watanabe, S. (1962). On some relations between the harmonic measure and the Lévy measure for a certain class of Markov processes. *J. Math. Kyoto Univ.* **2** 79--95.
14. *Internet Storm Center StormCast. Retrieved October 11, 2011, from* http://www.dshield.org

15. Jacod, J. (1975). Multivariate point processes: Predictable projection, Radon--Nikodým derivatives, representation of martingales. *Z. Wahrsch. Verw. Gebiete* **31** 235--253.
16. Jeanblanc, M. and Valchev, S. (2005). Partial information and hazard process. *Int. J. Theor. Appl. Finance* **8** 807--838.
17. Joyce, James, (2008) "Bayes' Theorem", *The Stanford Encyclopedia of Philosophy (Fall 2008 Edition)*, Edward N. Zalta (ed.), *Retrieved October 11, 2011,* from http://plato.stanford.edu/archives/fall2008/entries/bayes-theorem
18. Kay, R. (1977), "Proportional Hazard Regression Models and the Analysis of Censored Survival Data" *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, Vol. 26, No. 3 pp. 227-237 Blackwell Publishing.
19. Marti, K. (2008) "Computation of probabilities of survival/failure of technical, economic systems/structures by means of piecewise linearization of the performance function",Structural and Multidisciplinary Optimization, Vol35/3, Pp 225 - 244.
20. Newman, M.E.J., Strogatz, S.H., and Watts, D.J. (2001) "Random graphs with arbitrary degree distributions and their applications". Phys. Rev. E 64. paper 026118
21. Revuz, D. and Yor, M. (1999). *Continuous Martingale and Brownian Motion*, 3rd ed. Springer, Berlin.
22. Rogers, T. (1996) "*Type I and Type II Errors - Making Mistakes in the Justice System*" *Retrieved October 11, 2011, from* http://www.intuitor.com/statistics/T1T2Errors.html
23. *Survival/Failure Time Analysis*. Retrieved October 11, 2011,from http://www.statsoft.com/textbook/survival-failure-time-analysis
24. Therneau, T.; Sicks, J.; Bergstral, E. and Offord, J. (1994) "Expected Survival based on Hazard Rates", Technical Report No. 52, Mayo Foundation, *Retrieved October 11, 2011, from* http://mayoresearch.mayo.edu/biostat/upload/52.pdf
25. Thomson, I. (2007) *"Google warns of web malware epidemic" Retrieved October 11, 2011, from* http://www.securecomputing.net.au/News/81027,google-warns-of-web-malware-epidemic.aspx
26. Weisstein, Eric W. "*Bayes' Theorem" Retrieved October 11, 2011* from MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/BayesTheorem.html
27. Woller, J (1996) "The Basics of Monte Carlo Simulations" *Retrieved October 11, 2011,* from http://www.chem.unl.edu/zeng/joy/mclab/mcintro.html
28. Wright, C. (2007) "*The IT Regulatory and Standards Compliance Handbook: How to Survive Information Systems Audit and Assessments*" Syngress
29. Wright, C. & Zia, T. (2011)"A Quantitative Analysis into the Economics of Correcting Software Bugs" CISIS Spain
30. Zhu, H; Zhang, Y; Huo, Q & Greenwood, S; (2002)"Application of Hazard Analysis to Software Quality Modelling" Computer Software and Applications Conference, Annual International, p. 139, 26th Annual International Computer Software and Applications Conference