**IDEA GROUP PUBLISHING**
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

**ITB12704**

This paper appears in the book, *Emerging Trends and Challenges in Information Technology Management, Volume 1 and Volume 2* edited by Mehdi Khosrow-Pour © 2006, Idea Group Inc.

# Development of a Weighted Network Security Measure Based on a Software Reliability Model

Bong Gun Cho & Il-Yeol Song, College of Information Science & Technology, Drexel University, Philadelphia, PA 19104

Sung Y. Shin & Charlie Y. Shim, Dept of EE & Computer Science, South Dakota State University, Brooking, SD 57007

## ABSTRACT

A reliable security measure of a network system is important in evaluating possible attacks against computer networks. In this paper, we develop a security measure on reliability of a network using five security services. The five security services are integrity, confidentiality, authorization, availability, and authentication. Our security measure is based on Goel's Non-Homogenous Poisson Process model and is computed using the violated security service data collected by Computer Emergency Response Team at Carnegie Mellon University. The measure can be used as a design metric for the development of more secure systems.

## 1. INTRODUCTION

The proliferation of computer networks and especially the ubiquity of the Internet have made security one of the key areas in modern computing [1]. In order to eliminate potential devastating consequences, quantifying and measuring the "security level" of a system have become an important concern [3].

In this paper, we develop a security measure on reliability of a network. Our security measure is based on Goel's Non-Homogenous Poisson Process model and is computed using the violated security service data collected by Computer Emergency Response Team at Carnegie Mellon University.

Our security measure is computed using the weighted reliability value for each security service because each security service violation occurs very differently in real-world. Finally, systems may achieve many different degrees of security depending on the details of their construction in reality. To deal with more cases, we consider other factors such as network architecture that affect the security of systems [2].

## 2. RELATED WORK

In this section, we briefly survey approaches of existing software reliability models.

Many analytical models for software reliability measurement have been proposed. These are mainly based on failure histories:

(1) Time Between Failures Models: This approach considers the time between two consecutive failures, (i-1)th and (i)th failures.
(2) Failure Count Models: This model focuses on the number of failures during a specific time interval.
(3) Fault Seeding Models: The basic idea is to put a known number of faults in a program to find and unknown number of indigenous faults, which are assumed to still remain in the program.
(4) Input Domain Based Models: A set of test cases generated from an input distribution representing the operational usage of the program is used. An estimate of program reliability is obtained from the failures observed during the execution of the test cases.

Goel's NHPP model belongs to "Failure Count Model" and we use this model to develop our security measure.

The problems with all of the models mentioned is that they use some form of a failure history as recorded during the testing of the software being evaluated [2]. Our goal is to evaluate a security at the system design level, before implementation. Additionally, the above models have no weight. If we give weight to reliability of the security measure according to our statistical documentation, then the security measure will have a more accurate value. In Section 3, we develop a security measure including weighting and multiplying factors for network architecture.

## 3. DEVELOPING A SECURITY MEASURE

In this section, we present the derivation of our security measure. Our security measure consists of the following five security services: integrity, confidentiality, authorization, availability, and authentication. The five components are adopted from the work by Shim [2]. In addition, we also adopt the assumption that the basic shape of our security measure would be of the following form;

$$S(X_1, X_2, X_3, X_4, X_5) = C_0 + C_1*X_1 + C_2*X_2 + C_3*X_3 + C_4*X_4 + C_5*X_5$$

Where,

$X_i$ (i=1, 2, 3, 4, 5) = Characteristic value (or reliability) for integrity, confidentiality, authorization, availability, and authentication service, respectively.

$C_i$ (i=0, 1, 2, 3, 4, 5) = General constant ($C_0$) and proportional constant for the respective characteristics.

S represents the probability of a violation happening for the security services not supported by the system's countermeasures.

### 3.1. Getting a Characteristic Value $X_i$

In this section, we show how to compute the security measure S. The first step is to find a characteristic value $X_i$ for each security service.

To get a characteristic value, we will use the Table 1 collected by Computer Emergency Response Team at Carnegie Mellon University. We will also use Goel's NHPP model for software reliability.

The following Formulas (1), (2), (3), (4), and (5) are the formulas of Goel's NHPP model [4]. In this model, $N(t_n)$ and $\overline{N}(t_n)$ represent the cumulative number of software failures detected by time $t_n$ and the number of faults remaining in the system at time $t_n$, respectively [2].

*Table 1. Violated security service statistics (1997 ~ 2005)*

| Year | Integrity | Confidentiality | Authorization | Availability | Authentication | Total |
|---|---|---|---|---|---|---|
| 1997 | 2 | 2 | 24 | 1 | 0 | 29 |
| 1998 | 0 | 0 | 10 | 5 | 1 | 16 |
| 1999 | 1 | 1 | 14 | 5 | 1 | 22 |
| 2000 | 3 | 7 | 10 | 8 | 6 | 34 |
| 2001 | 5 | 3 | 32 | 12 | 5 | 57 |
| 2002 | 6 | 4 | 33 | 16 | 0 | 59 |
| 2003 | 5 | 4 | 25 | 19 | 1 | 54 |
| 2004 | 3 | 3 | 22 | 7 | 0 | 35 |
| 2005 | 1 | 5 | 21 | 13 | 0 | 40 |
| Total | 26 | 29 | 191 | 86 | 14 | 346 |

$$E(N(t_n)) = a[1 - \exp(-bt_n)] \quad (1)$$

$$E(\overline{N}(t_n)) = a[\exp(-bt_n)] \quad (2)$$

where *a* is the number of software errors to be eventually detected, and *b* is the occurrence rate of an error.

If the cumulative number of failures by time $t_n$ is given, we can estimate the value of *a* and *b* with the following formulas:

$$y_n = a[1 - \exp(-bt_n)] \quad (3)$$

$$a[t_n\{\exp(-bt_n)\}] = \sum_{i=1}^{n} [(y_i - y_{i-1})(t_i\exp(-bt_i) - t_{i-1}\exp(-bt_{i-1})) / (\exp(-bt_{i-1}) - \exp(-bt_i))] \quad (4)$$

where $y_1, y_2, \ldots y_n$ are the cumulative number of failures detected by times $t_1, t_2, \ldots, t_n$, respectively. The values of *a* and *b* are estimated with Equations (1) and (2), and the given values of $y_i$ and $t_i$ from the failure history data.

Formula (5) is the reliability for our systems:

$$R(x) = \exp[-a\{\exp(-bs) - \exp(-b(s+x))\}] \quad (5)$$

Where *s* is the time to (k-1) failures and *x* is the time between failures (k-1) and k.

To get the reliability of each security service, we need to insert the values in Table 1 into Formulas (3) and (4). The value of n is 9 because our security violation data is collected for nine years and the time unit is one year. We can have the following equations by inserting the values in Table 1 into the Formulas (3) and (4) and rearranging:

For integrity,

$$88e^{-10b} - 114e^{-9b} + 146e^{-b} - 120 = 0 \quad (6)$$

Let $e^{-b} = x$. Then, we have

$$88X^{10} - 114X^9 + 146X - 120 = 0 \ (0<X<1) \quad (7)$$

For confidentiality,

$$92X^{10} - 121X^9 + 169X - 140 = 0 \ (0<X<1) \quad (8)$$

For authorization,

$$695X^{10} - 886X^9 + 1024X - 833 = 0 \ (0<X<1) \quad (9)$$

For availability,

$$254X^{10} - 340X^9 + 520X - 434 = 0 \ (0<X<1) \quad (10)$$

For authentication,

$$65X^{10} - 79X^9 + 61X - 47 = 0 \ (0<X<1) \quad (11)$$

We can now find the characteristic value for each security service by inserting the values *a* and *b*, obtained by solving each of the Equations (7), (8), (9), (10), and (11), respectively, into the Formula (5).

We use **MATLAB** to solve Equations (7) to (11). The followings are *x* values for those equations above:

For integrity, $f(0.99421) \approx 0$.

For confidentiality, $f(0.99523) \approx 0$.

For authorization, $f(0.997215) \approx 0$

For availability, $f(0.99752) \approx 0$.

For authentication, $f(0.906025) \approx 0$

As the next step, we will get *a* and *b* values from each obtained x (= $e^{-b}$) value.

For integrity,

$x = 0.99421 \rightarrow b = 0.005807$

$a = 26 / (1 - e^{-9b}) \rightarrow a = 510.594843$

For confidentiality, b = 0.004782, a = 688.427300

For authorization, b = 0.002789, a = 7705.030457

For availability, b = 0.002483, a = 3891.402715

For authentication, b = 0.098688, a = 23.785132

In this step, we will get reliability for each security service such as integrity, confidentiality, authorization, availability, and authentication. To find those reliabilities, we will insert obtained *a* and *b* into Formula (5)**.** The above Table 2 shows the reliabilities for five security services for 1 month.

### 3.2. Making Formula for Security Measure S

In this section, we develop our security measure S using the data in Table 3 and MATLAB.

There are 32 possible combinations of security services that can be provided by countermeasures included in a system design because the security services provided in a system vary according to the countermeasures chosen for the system [2].

Our choice of S value for each system that provides a particular combination of security services is the probability of a violation happening for the security services not supported by the system's countermeasures [2]. For example, if a system's countermeasures support authorization, availability, and authentication, then $S = (y_{n1} + y_{n2}) / (y_{n1} + y_{n2} + y_{n3} + y_{n4} + y_{n5})$. $y_{ni}$ (i=1, 2, 3, 4, 5) is the cumulative number of integrity, confidentiality, authorization, availability, and authentication violations (respectively) found by time $t_n$.

Table 3 shows the set of data points created using this approach that we used to find the proportional constant for each security service. In Table 3, we only show 6 cases among the 32 cases. Each of the integrity,

*Table 2. Reliability of each security service for 1 month*

| | Integrity | Confidentiality | Authorization | Availability | Authentication |
|---|---|---|---|---|---|
| 1 month | 0.791074 | 0.769286 | 0.175284 | 0.457409 | 0.922994 |

*Table 3. Data set for regression analysis (for 1 month)*

| | Security | Integrity | Confidentiality | Authorization | Availability | Authentication |
|---|---|---|---|---|---|---|
| 1 | 1 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 2 | 0.924855 | 0.791074 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 3 | 0.916185 | 0.000000 | 0.769286 | 0.000000 | 0.000000 | 0.000000 |
| 4 | 0.447977 | 0.000000 | 0.000000 | 0.175284 | 0.000000 | 0.000000 |
| 5 | 0.751445 | 0.000000 | 0.000000 | 0.000000 | 0.457409 | 0.000000 |
| 6 | 0.959538 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.922994 |

confidentiality, authorization, availability, and authentication columns represents the reliability for each security service, respectively.

Using the Table 3 and **MATLAB**, we can have the following formula for our security measure S:

$$S(X_1, X_2, X_3, X_4, X_5) = 1 - 0.09499111*X_1 - 0.10895168*X_2 - 3.14930627*X_3 - 0.54339770*X_4 - 0.04383777*X_5$$

### 3.3. Giving Weight to each Reliability for Security Measure S

In this section, we give a different weight to reliability of the each security service in the measure S obtained in Section 3.2 because each security service violation occurred very differently in real-world. The following is one way to find an appropriate weight.

As we can see in Table 1, the total violated security service number is 346. We can also know that the average number of violations per security service is approximately 69 (346 / 5 = 69.2). Therefore, if each security service has 69 violated security service cases, we give weight = 1 to our each reliability. It means that we multiply each reliability by weight = 1.

- 68 violated case: -1 ➔ 69 / 68 ≈ 1.014➔ weight = 1.014
- 69 violated case: 0 ➔ 69 / 69 = 1.000 ➔ weight = 1
- 70 violated case: +1 ➔ 69 / 70 ≈ 0.9857➔ weight = 0.9857

However, when we apply ±0.014 per one case over or below 69 cases, the reliability (0<R<1) for authentication is over 1. Therefore, we can't use this weight and have to find other appropriate weight.

Here is the final and reasonable weight:

- 0.014 * 0.5 = 0.007
- 0.007 * 0.5 = 0.0035
- 0.0035 * 0.5 = 0.00175
- 0.00175 * 0.5 = **0.000875**➔ it has no problem with the range of reliability for all security service
- 68 violated case: -1 ➔ weight = 1 + 0.000875 = 1.000875
- 69 violated case: 0 ➔ weight = 1 + 0.0000 = 1.0000
- 70 violated case: +1 ➔ weight = 1 – 0.000875 = 0.999125

In violated security service statistics from 1997 to 2005,

- Integrity: 26 violated cases ➔ -43 (from 69 violated cases)
- Confidentiality: 29 violated cases ➔ -40
- Authorization: 191 violated cases ➔ +122
- Availability: 86 violated cases ➔ +17
- Authentication: 14 violated cases ➔ -55

The followings are examples for integrity and authorization case (for 1 month):

*Table 4. Reliability for each security service after weight for 1 month*

| | Integrity | Confidentiality | Authorization | Availability | Authentication |
|---|---|---|---|---|---|
| 1 month | 0.820838 | 0.796211 | 0.156572 | 0.450605 | 0.967413 |

For integrity (1 month case),

Current reliability for integrity = 0.791074

0.000875 * 43 = 0.037625 ➔ weight = 1 + 0.037625 = 1.037625

Now we multiply current reliability for integrity (0.791074) by weight (1.037625).

➔0.791074 * 1.037625 = 0.820838

### 3.4. Improving the Security Measure

Systems may achieve many different degrees of security depending on the details of their construction. To deal with a number of cases, we must consider other factors such as the network architecture that affect the security of systems [2].

We note that, as we use encryption in a higher network layer, we can achieve more security. By applying this idea, we can infer that security mechanisms for a system can achieve different degrees of security according to the layers at which each of the security services is provided [2].

Figure 1 below shows multiplying factor for each layer based on TCP/IP model. The multiplying factor is a number by which the reliability for a security service provided is multiplied to differentiate the same kinds of security services that are provided at different network layers.

Since the TCP/IP model has 5 layers, we decrease the multiplying factor by 0.2 (= 1/5) as the layer goes down one level.

If a service is provided at the application layer, we multiply its reliability $X_i$ by 1. This means that we assign 100% of the reliability that we measured to the service.

In summary, our final security measure is stated as follows:

$$S(X_1, F_1, X_2, F_2, X_3, F_3, X_4, F_4, X_5, F_5) = 1 - 0.09154669X_1*F_1 - 0.10526732X_2*F_2 - 3.52568148X_3*F_3 - 0.55160285X_4*F_4 - 0.04182495X_5*F_5$$

where,

$X_i$ (i = 1, 2, 3, 4, 5) = reliability for integrity, confidentiality, authorization, availability, and authentication, respectively

Each $F_1$, $F_2$, $F_3$, $F_4$, and $F_5$ represents the multiplying factor for each security service as explained above.

## 4. CONCLUSION

In this paper, we developed an empirical network security measure based on software reliability model. Our security measure is based on Goel's Non-Homogenous Poisson Process model and was computed using the violated security services data collected by Computer Emergency Response Team at Carnegie Mellon University. In developing our security measure, we used different weights to each security service because each security service violation occurred very differently in real-world.

*Figure 1. TCP/IP Model and multiplying factor*

| TCP/IP Model | Multiplying Factor |
|---|---|
| Application | 1 |
| TCP | 0.8 |
| IP | 0.6 |
| Data Link | 0.4 |
| Physical | 0.2 |

The security measure in this paper enables us to compare two systems and determine which is more secure. Our measure is not an absolute measure but a relative measure. Since weighted, this measure reflects the current tendency of security violations more accurately. Thus, system developers can develop more secure network by using our measure.

## REFERENCE

[1] Giampaolo Bella, Ronaldo Menezes, and James Whittaker, "Editorial Message Special Track on Computer Security," Proceeding of the 2002 ACM Symposium on Applied Computing, ACM Press, 2002, pp. 194-195.

[2] Yong-Sang Shim, Developing a Probabilistic Security Measure Using a Software Reliability Model, Ph.D. dissertation, University of Wyoming, 2001.

[3] A.L. Goel et al., "A time dependent error detection rate model for software reliability and other performance measures," IEEE Transactions on Reliability, R-28, 1979, pp. 206-211.

[4] A.L. Goel et al., Software reliability modeling and estimation techniques, Rep. RADC-TR 82, 1982, pp. 263.