

An Input–Output Measurable Design for the Security Meter Model to Quantify and Manage Software Security Risk

Mehmet Sahinoglu, *Senior Member, IEEE*

Abstract—The need for information security is self-evident. The pervasiveness of this critical topic requires primarily risk assessment and management through quantitative means. To do an assessment, repeated security probes, surveys, and input data measurements must be taken and verified toward the goal of risk mitigation. One can evaluate risk using a probabilistically accurate statistical estimation scheme in a quantitative security meter (SM) model that mimics the events of the breach of security. An empirical study is presented and verified by discrete-event and Monte Carlo simulations. The design improves as more data are collected and updated. Practical aspects of the SM are presented with a real-world example and a risk-management scenario.

Index Terms—Assessment, cost, countermeasure, data, management, probability, quantity, reliability, risk, security, simulation, statistics, threat, vulnerability.

I. INTRODUCTION—WHY MEASURE AND ESTIMATE THE INPUTS IN THE SM MODEL

QUANTITATIVE risk measurements are needed to objectively compare alternatives and calculate monetary figures for budgeting and for reducing or minimizing the existing risk. Security meter (SM) design provides these conveniences in a quantitative manner that is much desired in the security world [1], [7]–[11]. This is a follow up to [1] to create a simple statistical input–output design to estimate the risk model’s parameters in terms of probabilities. In pursuit of a practical and accurate statistical design, security breaches will be recorded, and then, the model’s input probabilities will be estimated using the equations that were developed. Undesirable threats that take advantage of hardware and software weaknesses or vulnerabilities can impact the violation and breakdown of availability (readiness for usage), integrity (accuracy), confidentiality, and nonrepudiation, as well as other aspects of software security such as authentication, privacy, and encryption [2]. Other methods such as Attack Trees [3], [4], Time-to-Defeat [5], and qualitative models [6] are only deterministic. Therefore, we must collect data for malicious attacks that have been prevented or not prevented [7]–[9]. Fig. 1 shows that the constants are the utility cost (asset) and criticality constant (between 0 and 1), whereas the probabilistic inputs are vulnerability, threat, and lack of countermeasure (LCM) of all risks between 0 and 1. The

residual risk (RR: as in Fig. 2) and expected cost of loss (ECL) are the outputs obtained using (1)–(3). Fig. 3 will illustrate a software solution.

The black box in Fig. 1 leads to the probabilistic tree diagram of Fig. 2 to do the calculations.

Equations (1)–(3) summarize Figs. 1 and 2 from input to output. Suppose an attack occurs, and it is recorded. At the very least, we need to come up with a percentage of nonattacks and successful (from the adversary’s viewpoint) attacks. Out of 100 such attempts, the number of successful attacks will yield the estimate for the percentage of LCM. We can then trace the root of the cause to the threat level backward in the tree diagram. Let us imagine that the anti-virus software did not catch it, and a virus attack occurs, which reveals the threat exactly. As a result of this attack, whose root threat is known, the e-mail system may be disabled. Then, the vulnerability comes from the e-mail itself. This way, we have completed the “line of attack” on the tree diagram, as illustrated in Fig. 2. Out of 100 such cyberattacks, which maliciously harmed the target cyberoperation in some manner, how many of them were not prevented or countermeasured by, e.g., smoke detectors or generators or antivirus software or firewalls installed? Out of those that are not prevented by a certain CM device, how many of them were caused by threat 1 or 2, etc., of certain vulnerability? We can then calculate the percentage of vulnerability A, B, or C. The only way wherein we can calculate the count of CM preventions is by doing either of the following: a) guessing a healthy estimator of an attack ratio, like 2% of all attacks are prevented by CM devices or b) using a countermeasuring device to detect a probable attack prematurely. The following equation computes the RRs for each activity in Table II for each leg:

$$RR = \text{Vulnerability} \times \text{Threat} \times \text{LCM}. \quad (1)$$

II. SIMPLE CASE STUDY FOR THE PROPOSED SM

The suggested vulnerability (weakness) values vary between 0.0 and 1.0 (or between 0% and 100%) to add up to one. In a probabilistic sample space of feasible outcomes of the random variable of “vulnerability,” the sum of probabilities adds up to one. This is like the probabilities of the faces of a die, such as 1 to 6, totaling to one. If a cited vulnerability is not exploited in reality, then it cannot be included in the model or Monte Carlo (MC) simulation study. Vulnerability has from one to several threats to trigger the existing vulnerability. A threat is defined

Manuscript received April 27, 2007; revised August 17, 2007.

The author is with the Department of Computer Science, Troy University, Montgomery, AL 36103 USA (e-mail: mesa@troy.edu).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIM.2007.915139

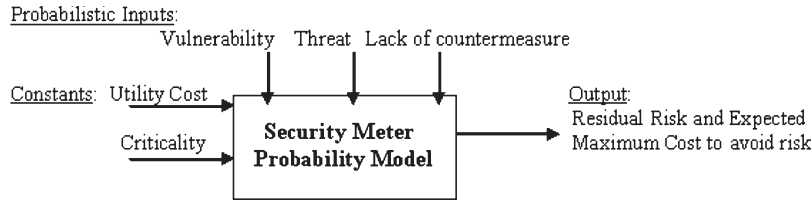


Fig. 1. Quantitative SM model of probabilistic and deterministic inputs and outputs.

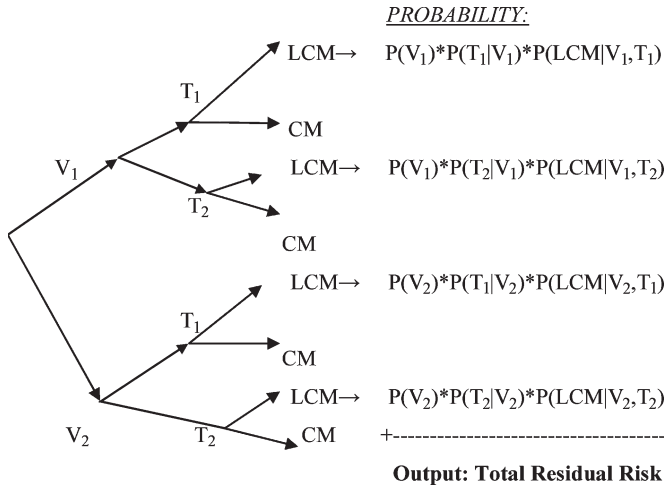


Fig. 2. Simple tree diagram for two threats per each of the two vulnerabilities.

as the probability of the exploitation of some vulnerability or weakness within a specific time frame.

Each threat has a countermeasure (CM) that ranges between 0 and 1 (with respect to the first law of probability) whose complement gives the LCM. The binary CM and LCM values should add up to one, keeping in mind the second law of probability. The security risk analyst can define, for instance, a network server v_1 as a vulnerability located in a remote unoccupied room in which a threat t_{11} , such as individuals without proper access, or a fire t_{12} could result in the destruction of assets if not countermeasured by items such as a motion sensor CM_{11} or a fire alarm CM_{12} , respectively. System criticality, which is another constant that indicates the degree of how critical or disruptive the system is in the event of an entire loss, is taken to be a single value that corresponds to all vulnerabilities with a value ranging from 0.0 to 1.0, or from 0% to 100%. Criticality is low if the RR is of little or no significance, such as the malfunctioning of an office printer, but in the case of a nuclear power plant, criticality is close to 100%. Capital (investment) cost is the total expected loss in monetary units (e.g., dollars) for the particular system if it is completely destroyed and can no longer be utilized, excluding the shadow costs, had the system continued to generate added value for the system. The following example in Tables I and II is studied to illustrate the application of the SM model [1]:

$$\begin{aligned}
 \text{Final Risk} &= \text{RR} \times \text{Criticality} \\
 &= 0.5 \times 0.5 \\
 &= 0.25
 \end{aligned} \tag{2}$$

where Criticality = 0.5, and

$$\begin{aligned}
 \text{Expected Cost of Loss (ECL)} &= \text{Utility Cost} \times \text{Final Risk} \\
 &= \$1000 \times 0.25 \\
 &= \$250
 \end{aligned} \tag{3}$$

for Utility Cost = \$1000.

III. BAYESIAN TECHNIQUE TO PRIORITIZE SOFTWARE MAINTENANCE USING THE SM

An SM's mathematical accuracy is verified by an MC simulation study in Fig. 3 for Tables I and II. Five thousand runs, in which one run is displayed in Fig. 3 for illustration purposes, are conducted by generating uniformly distributed random variables from each vulnerability, threat, or CM. Then, the SM method takes effect in multiplying the conditional probabilities at each branch with respect to (1), Figs. 1 and 2, and Tables I and II to calculate the RRs and finally sum them up for the total RR. The average of a selected total number of runs such as $5000 \times 1000 = 5$ million will be the final MC result. Then, (2) and (3) are used to reach the final risk and cost. Fig. 3 displays the input data for $v(a, b)$, $t(a, b)$, and $CM(a, b)$, which are taken to be uniformly distributed, i.e., $U(a, b)$. The lower and upper bound values for the last window in the case of the second vulnerability or second set of threats in this example are left blank, as the software will complement them to 1.0 to obey the fundamental probability laws. Fig. 3 shows the final risk and the ECL. By using a single shot for one simulation trial, as shown in Fig. 3, assuming that it is a hypothetical example, let us apply a Bayesian formula to determine the vulnerability that needs the most care for maintenance. Let us now ask the Bayesian question in relevance to our maintenance problem. Given that the office computer has R for risk, what is the probability that it failed due to CPU (fire/system down) or E-Mail (virus/hacking) vulnerability? We need to find the following Bayesian probabilities [10]:

$$P(\text{Fire} | R) = 0.058375 / 0.468838 = 0.1246 \tag{4}$$

$$P(\text{System Down} | R) = 0.070589 / 0.468838 = 0.1506 \tag{5}$$

$$P(\text{Virus Attack} | R) = 0.154611 / 0.468838 = 0.3298 \tag{6}$$

$$P(\text{Hacking Attack} | R) = 0.185262 / 0.468838 = 0.3950. \tag{7}$$

From these Bayesian posterior probabilities, it is obvious that the posterior risk (R) due to physical failures of the first vulnerability is $0.1246 + 0.1506 = 0.2752$, or 27.52%. On the other hand, the prior contribution of the physical failures at the

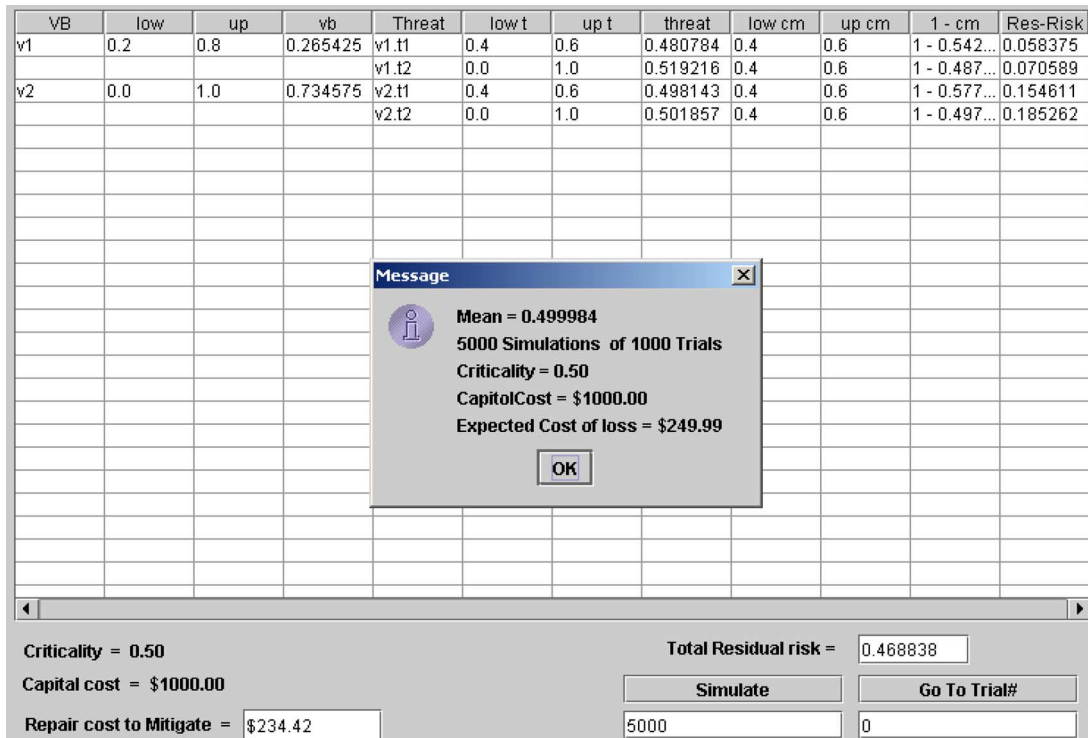


Fig. 3. Simulation of Tables I and II from Fig. 2 with software from www.areslimited.com.

TABLE I
VULNERABILITY-THREAT-CM SPREADSHEET FOR AN OFFICE COMPUTER

Vulnerability	Threat	Countermeasure
CPU	1. Fire	1. Smoke-detectors
	2. System Down	2. In-house generator
E-Mail	1. Virus	1. Installed anti-virus software
	2. Hacking	2. Installation of firewall

TABLE II
INPUT DATA (EXPECTED VALUES) AND CALCULATED RISK FOR TABLE I AND FIG. 2

0.5	0.5	0.5	
		0.5	0.125
	0.5	0.5	
		0.5	0.125
0.5	0.5	0.5	
		0.5	0.125
	0.5	0.5	
		0.5	0.125
<i>Summed Residual Risk: 0.500</i>			

very beginning stage was less: 0.2654 or 26.54% in Fig. 3. On the other hand, the prior contribution due to e-mail vulnerability was 0.7345, or 73.45%. The resulting posterior contribution was $0.3298 + 0.3950 = 0.7248$, or 72.48%. What this means is that although malicious causes of the second vulnerability con-

stitute 73.45% of the totality of failures, these causes generate 72.48% of the risks. More care for CPU maintenance is required on the first vulnerability than the second one. In addition, the threat of system down is more severe than that of the fire threat in the first (CPU) vulnerability.

IV. ANALYTICAL FORMULAS ON HOW TO CALCULATE THE INPUT PARAMETERS

We will apply the relative frequency (based on the law of large numbers) approach [12]. Let X be the total number of saves or prevented crashes by a CM device within a unit time such as a month or a year. Let Y be the number of unpreventable crashes that caused a security breach for different reasons. Let us assume that a track analysis showed the following in an all-double $2 \times 2 \times 2$ SM model like that in Fig. 2 and Table I: Out of Y number of crashes, there were $Y_{11}(v_1, t_1)$ counts due to threat t_1 and $Y_{12}(v_1, t_2)$ counts due to threat t_2 , all stemming from vulnerability 1. Further, it was tracked that there were $Y_{21}(v_2, t_1)$ crashes due to threat t_1 and $Y_{22}(v_2, t_2)$ crashes due to threat t_2 , all stemming from vulnerability 2. One could generalize this to $Y(v_i, t_j) = Y_{ij}$ caused by the i th vulnerability and its j th threat. Similarly, one assumes that there were $X(v_i, t_j) = X_{ij}$ “saves,” which could have happened on the i th vulnerability and the j th threat. Therefore

$$Y(\text{saves}) = \sum_i \sum_j Y(v_i, t_j) = \sum_i \sum_j Y_{i,j} \quad (8)$$

$$X(\text{crashes}) = \sum_i \sum_j X(v_i, t_j) = \sum_i \sum_j X_{i,j} \quad (9)$$

where $i = 1, 2, \dots, I$, and $j = 1, 2, \dots, J$.

Then, we can find the probability estimates for the threats $P(v_i, t_j)$ by taking the ratios as follows:

$$P_{ij} = \frac{X_{ij} + Y_{ij}}{Y_i + X_i}, \quad \text{for a given “}i\text{”}$$

$$Y_i = \sum_j Y_{ij}, \quad X_i = \sum_j X_{ij}, \text{ for } j = 1, 2, \dots, J. \quad (10)$$

It follows that for the probabilities of vulnerabilities, we have

$$P_i = \frac{\sum_j (X_{ij} + Y_{ij})}{\sum_i \sum_j (X_{ij} + Y_{ij})}$$

for $i = 1, 2, \dots, I$ and $j = 1, 2, \dots, J$. (11)

Last, the probability of LCM, i.e., $P(\text{LCM}_{ij})$, for $i = 1, 2, \dots, I$ and $j = 1, 2, \dots, J$ is estimated as

$$P(\text{LCM}_{ij}) = \frac{Y_{ij}}{(Y_{ij} + X_{ij})}, \quad \text{for a given “}i\text{” and “}j\text{”} \quad (12)$$

$$P(\text{CM}_{ij}) = 1 - P(\text{LCM}_{ij}). \quad (13)$$

V. NUMERICAL EXAMPLE ON THE INPUT-MEASURABLE DESIGN FOR THE SM MODEL

Let there be a double-vulnerability and double-threat and CM & LCM setup ($2 \times 2 \times 2$) such as that shown in Fig. 2 and

Table I. Note that the data are obtained from an MC study in the author’s class [13]. Then

X (total number of detected anomalies :

crash preventions) = approximately 1/day or 360/year.

Let $X_{11} = 98$, $X_{12} = 82$, $X_{21} = 82$, and $X_{22} = 98$. Then

Y (total number of undetected anomalies :

crashes not prevented) = approximately 10/year.

Let $Y_{11} = 2$, $Y_{12} = 3$, $Y_{21} = 3$, and $Y_{22} = 2$. By implementing (8)–(13), we arrive at P_{11} (threat 1 risk for vulnerability 1) = $(X_{11} + Y_{11}) / (X_{11} + Y_{11} + X_{12} + Y_{12}) = 100 / 185 = 0.54$ and P_{12} (threat 2 risk for vulnerability 1) = $(X_{12} + Y_{12}) / (X_{11} + Y_{11} + X_{12} + Y_{12}) = 85 / 185 = 0.46$. Similarly, P_{21} (threat 1 risk for vulnerability 2) = $(X_{21} + Y_{21}) / (X_{21} + Y_{21} + X_{22} + Y_{22}) = 85 / 185 = 0.46$ and P_{22} (threat 2 risk for vulnerability 2) = $(X_{22} + Y_{22}) / (X_{21} + Y_{21} + X_{22} + Y_{22}) = 100 / 185 = 0.54$. Risk of vulnerability 1 is given by $P_1 = (X_{11} + Y_{11} + X_{12} + Y_{12}) / (X_{11} + X_{12} + X_{21} + X_{22} + Y_{11} + Y_{12} + Y_{21} + Y_{22}) = 185 / 370 = 0.5$. Risk of vulnerability 2 is given by $P_2 = (X_{21} + Y_{21} + X_{22} + Y_{22}) / (X_{11} + X_{12} + X_{21} + X_{22} + Y_{11} + Y_{12} + Y_{21} + Y_{22}) = 185 / 370 = 0.5$.

The probabilities of “LCM” and “CM” for the v – t pairs in Fig. 1 are given as follows.

$$P(\text{LCM}_{11}) = (Y_{11} / Y_{11} + X_{11}) = 2 / 100 = 0.02, \text{ and hence,}$$

$$P(\text{CM}_{11}) = 1 - 0.02 = 0.98.$$

$$P(\text{LCM}_{12}) = (Y_{12} / Y_{12} + X_{12}) = 3 / 85 = 0.035, \text{ and hence,}$$

$$P(\text{CM}_{12}) = 1 - 0.035 = 0.965.$$

$$P(\text{LCM}_{21}) = (Y_{21} / Y_{21} + X_{21}) = 3 / 85 = 0.035, \text{ and hence,}$$

$$P(\text{CM}_{21}) = 1 - 0.035 = 0.965.$$

$$P(\text{LCM}_{22}) = (Y_{22} / Y_{22} + X_{22}) = 2 / 100 = 0.02, \text{ and hence,}$$

$$P(\text{CM}_{22}) = 1 - 0.02 = 0.98.$$

Let us place the aforementioned estimated input values in Fig. 2 for the model to calculate the risk in Fig. 4.

Therefore, once the probabilistic model from the empirical data is built, as stated earlier in this paper, which should verify the final results, one can forecast or predict any taxonomic activity whether the number of vulnerabilities, threats, or crashes exists. For the above study, the total number of crashes is 10 out of 370, which gives a ratio of $10 / 370 = 0.0270$. This agrees with the risk calculations in Fig. 4 if you add up risks. Hence, if after building this probabilistically accurate model, one in a different setting or year can predict what will happen for a given explanatory set of data. If a clue gives us 500 episodes of vulnerabilities in V_1 , then, by the avalanche effect, we can find all the other blanks, such as for $V_2 = 500$. Then, $0.54(500) = 270$ of T_1 and $0.46(500) = 230$ of T_2 . Out of 270 T_1 episodes, $0.02(270) = 5.4$ for LCM, yielding 5 (rounded off) “crashes.” Thus, antivirus or firewalls have enabled 265 (rounded off) preventions or “saves.” Again, for T_2 in V_1 , $(0.035)230 = 8$ (rounded off) “crashes” and $0.965(230) = 222$ (rounded off) “saves,” all shown in Table III [12]. If the asset is \$2500 and criticality is 0.4, ECL is $\text{RR} \times \text{Criticality} \times \text{Asset} = 0.027 \times 0.4 \times \$2500 = \$27$, as in [1].

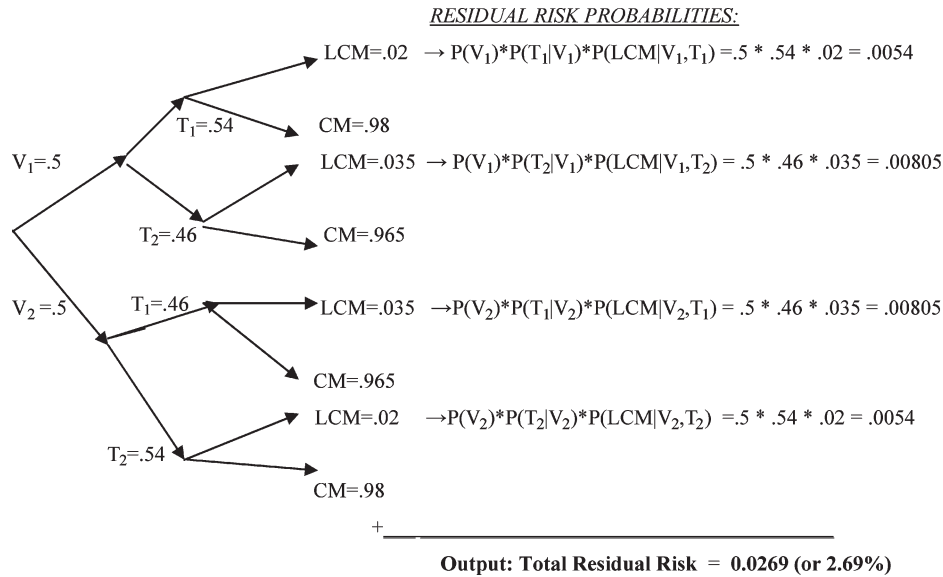


Fig. 4. Simple tree diagram for two threats per two vulnerabilities using Section V inputs.

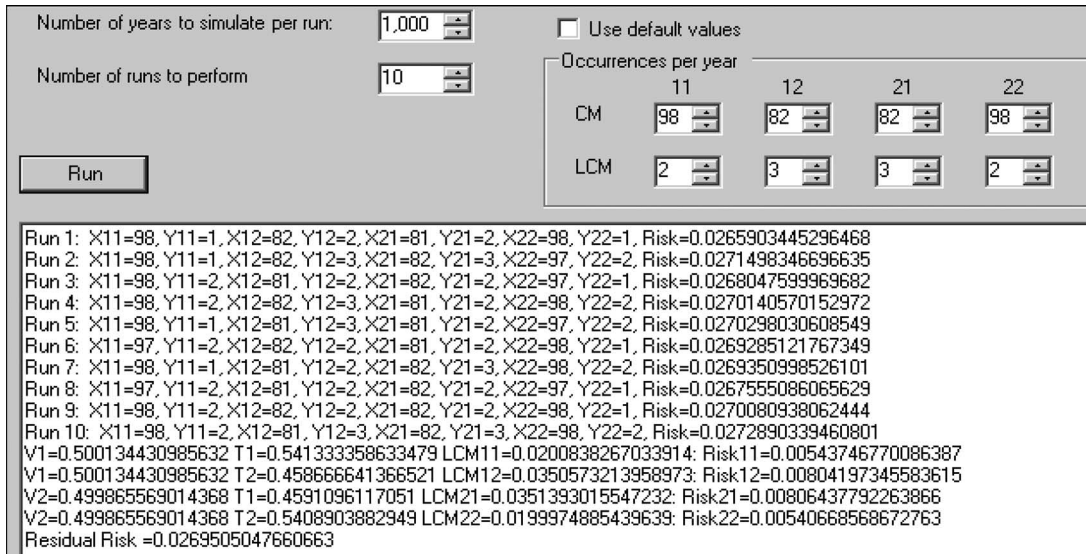


Fig. 5. Discrete-event simulation results of the 2 × 2 × 2 SM sample design.

VI. RANDOM NUMBER SIMULATIONS TO VERIFY THE ACCURACY OF THE SM DESIGN

A. Discrete Event (Dynamic) Simulation Method

The analyst is expected to simulate a component, like a server, from the beginning of the year (like 1/1/2006) until the end of 1000 years (1/1/3006) in an 8 760 000-h period, with a life cycle of “crashes” or “saves.” The input data are supplied in Fig. 5 for the simulation of random deviates. At the end of this planned time period, the analyst will fill in the elements of the tree diagram for the 2 × 2 × 2 SM model, as in Fig. 4. Recall that the rates are the reciprocals of the means for an assumption of negative exponential probability density function to represent the distribution of time-to-crash. For example, if rate = (98/8760) h⁻¹, then the mean time to crash is 8760/98 = 89.38 h. Use the same input data in Section V as in Fig. 5 [13].

B. MC (Static) Simulation Method

Using the identical information in Section VI-A, the analyst is expected to use the principles of MC simulation to simulate the 2 × 2 × 2 SM, as in Fig. 4. One employs the Poisson distribution for generating rates for each leg in the tree diagram of the 2 × 2 × 2 model, as in Figs. 4 and 6. The rates are given as the count of saves or crashes per annum. The necessary rates of occurrence for the Poisson random value generation are already given in the empirical data example earlier in Section V. For each SM realization, get a risk value, and then, average it over n = 10 000 in 1000 increments. When averaged over n = 1000 runs, one should aim to get the same value as in Figs. 4 and 5. The same data as in Section V is used in Fig. 6, which tabulates the MC simulation as an alternative to the discrete-event simulation in Fig. 5. They both reach the same results with the increase of the simulation runs.

TABLE III
ESTIMATION OF THE MODEL PARAMETERS, GIVEN THE TOTAL NUMBER OF ATTACKS, i.e., THE TAXONOMY OF SAVES AND CRASHES IN THE “370” AND “1000” ATTACKS SCENARIO AS IN THE SECTION V EXAMPLE

Total Attacks	VB	Attacks	%	crashes	saves	Threat	events	%	crashes	saves	Risk	Post Pct	Post vb
370	v1	185	50.00	5	180	v1.t1	100	54.05	2	98	0.005405	20.00	
						v1.t2	85	45.95	3	82	0.008108	30.00	0.500000
	v2	185	50.00	5	180	v2.t1	100	54.05	2	98	0.005405	20.00	
						v2.t2	85	45.95	3	82	0.008108	30.00	0.500000

Results Table

Total Attacks	VB	Attacks	%	crashes	saves	Threat	events	%	crashes	saves	Risk	Post Pct	Post vb
1000	v1	500	50.00	14	486	v1.t1	270	54.05	5	265	0.005005	19.26	
						v1.t2	230	45.95	8	222	0.007991	30.74	0.500000
	v2	500	50.00	14	486	v2.t1	270	54.05	5	265	0.005005	19.26	
						v2.t2	230	45.95	8	222	0.007991	30.74	0.500000

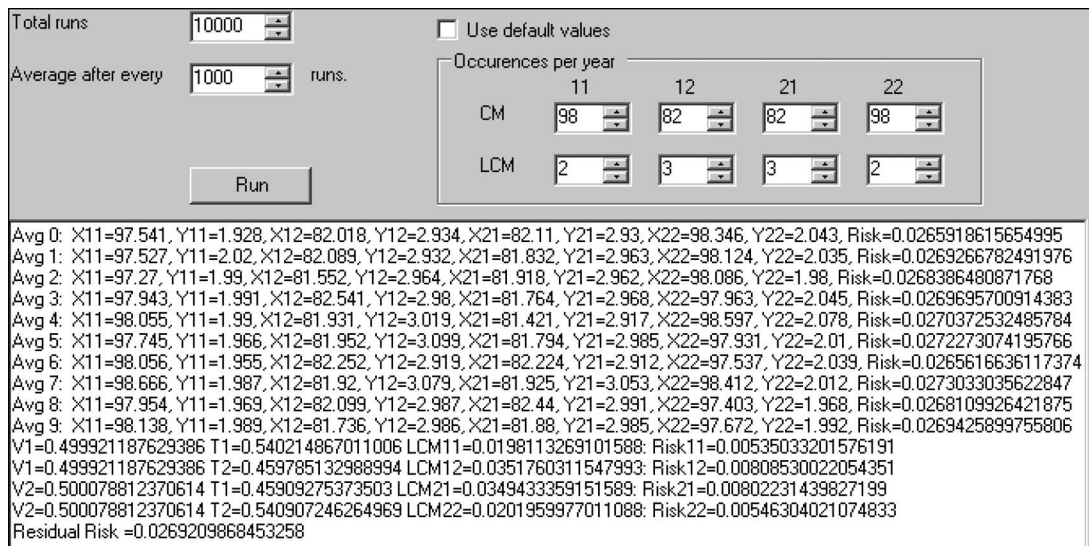


Fig. 6. MC simulation results of the 2 × 2 × 2 SM sample design.

VII. REAL-WORLD EXAMPLE TO IMPLEMENT THE SM DESIGN USING SURVEY RESULTS

In a recent field study by Wentzel [25] to implement the security model, Computer Security Institute/Federal Bureau of Investigation, Deloitte, and Pricewaterhouse survey results were evaluated regarding the security concerns at the University of Virginia School of Continuing and Professional Studies Northern Virginia Regional Center (from now on, Center), located in a stand-alone 105 000-square-foot four-story building adjacent to the West Falls Church Metro train station. There are seven servers that are located at the facility. The Center’s Senior Network Administrator estimated the servers in May 2006 to be worth \$8000 each. This cost estimate includes all Dell hardware, Windows 2003 server software and licensing, accessories (new cables and racks), and personnel time, but not the Cisco firewall or backup power supplies. With our goal of a general risk assessment in mind and the quantitative SM method selected, the next step was to determine the scope of the risk assessment project. One original goal of this quantitative risk assessment [25]–[29] was to follow up on a recently completed qualitative assessment and, then, to compare the two sets of results. In the qualitative Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE-S) risk

assessment, critical assessments were found to be the *servers* and the *instructors* at the Center. Later, however, the analyst found three sources of national surveys only on the physical assets (dropping the instructor), no matter how difficult due to widely spread notion that organizations do not like to admit security breaches. The surveys pertained to similar data servers that contain the material associated with marketing business function, which was the primary independent business mission. Survey results were used as source for probability data to compile the vulnerability, threat, and CM risks in the SM (see Tables IV–VII).

Therefore, a criticality rating of 0.4 is selected in this example for an asset of \$8000 for each server to be used in the SM calculations of the RR in Table VIII. Note that the symbols used in the source data, such as CM_{ij} for threat t_{ij} from vulnerability v_i , wherein, for this Center example, $i = 1, j = 1, 2, 3, 4, i = 2, j = 1, 2, 3$, and, finally, $i = 3, j = 1, 2$ are all illustrated in Table VIII for the risk assessment and management stages.

VIII. RISK MANAGEMENT EXAMPLE FOR THE “COMPUTER CENTER” CASE STUDY

Once the SM is applied and the RR is calculated based on the data in Section VII, the security risk manager would want to

TABLE IV
SM PROBABILITY DATA FOR CM ACTIONS USED IN TABLE VIII

CSI / FBI Survey Countermeasure [26]	% of Respondents Positively Reporting Them
Firewalls	97 (= CM ₁₃)
Anti-Virus Software	96 (= CM ₂₃)
Security Audits	80 (= CM ₁₄)
Intrusion Detection Systems (IDS)	72 (= CM ₃₁)
Security Awareness Policy Training	70 (= CM ₁₁)
Server-Based Access Control Lists (ACL)	70 (= CM ₃₂)
Encryption For Data In Transit	68 N/A or redundant, not used in the final table
Reusable Account/Login Passwords	52 N/A or redundant, not used in the final table
Encrypted Files	46 (= CM ₃₃)
Smart Cards/Other One-Time Password Tokens	42 (= CM ₁₂)
Public Key Infrastructure	35 (= CM ₂₁)
Intrusion Prevention Systems	35 (= CM ₂₂)
Biometrics	15 N/A or redundant, not used in the final table

TABLE V
SM PROBABILITY DATA FOR THREATS PERTAINING TO EACH VULNERABILITY USED IN TABLE VIII

CSI / FBI [26] Survey Threats and CIO [27] / Pricewaterhouse Survey Threats [28]	% of Respondents Positively Reporting Them
Virus	76 (= T ₂₃)
Malicious Code	59 (= T ₃₂)
Insider Abuse of Net Access	48 (= T ₁₁)
Laptop/Mobile Theft	48 N/A or redundant, not used in the final table
Unauthorized Access to Information	32 (= T ₃₁)
Denial of Service (DOS)	32 (= T ₁₃)
Abuse of Wireless Network and Web Site Defacement	22 (= T ₂₁)
System Penetration	16 (= T ₁₂)
Theft of Proprietary Information	9 (= T ₃₃)
Misuse of Public Web Application	5 N/A or redundant, not used in the final table
Financial or Telecom Fraud	4 (= T ₁₄)
Sabotage	2 (= T ₂₂)

TABLE VI
SM PROBABILITY DATA FOR THE THREE DISJOINT VULNERABILITIES USED IN TABLE VIII

Deloitte Survey Vulnerabilities [28]	% of Respondents Positively Reporting Them
Internal Security Breach only	35 (= V ₁)
External Security Breach only	26 (= V ₂)
Both Internal and External Security Breach	39 (= V ₁ and V ₂)

calculate as to how much s(he) needs to spend on improving the CMs (firewall, intrusion detection system, virus protection, etc.) to mitigate risk. On the expense side, one has a cost accrued per 1% unit improvement on the CM, which is the only parameter of the SM that one may voluntarily monitor. The average cost C per 1% must be known to cover personnel, equipment, and all devices. On the positive side, the ECL will decrease with a gain of ΔECL , while the software/hardware CM improvements are

added on. The breakeven point is where the pros and cons are equal, guiding the security manager as to how one can move to a more advantageous state from the present. In the *Base Server* of Fig. 7, the organizational policy of mitigating the RR from 26.04% down to 10% ($\leq 10\%$) in the *Improved Server* has been satisfactorily illustrated. By solving for a breakeven cost $C = \$5.67$ per unit percent improvement in the CM, for each improvement action, such as increasing from 70% to 100%

TABLE VII
ASSET CRITICALITY RATING FOR THE SM DESIGN FOR AN ASSET OF \$8000

Criticality Definition [29]	Value Rating Factor
Asset's Loss has negligible impact on Center's mission	0.0
Asset's Loss has minor impact on Center's mission	0.2
Asset's Loss has moderate impact on Center's mission	0.4 (selected in this example)
Asset's Loss has significant impact on Center's mission	0.6
Asset is mission-critical to the Center. Loss would have serious impact on Center's Mission.	0.8
Asset is mission-essential to the Center. Center could not absolutely carry out mission without it.	1.0

TABLE VIII
SM PROBABILITY TABLE FOR A PRODUCTION SERVER AT THE CENTER USING TABLES IV-VI

Vulnerability	Threat	Countermeasure
V ₁ = 0.35 (Internal Security Breach Only)	T ₁₁ = 0.48 (Internal Abuse of Network Access)	CM ₁₁ = 0.70 (Security Awareness Policy Training) LCM ₁₁ = 0.30 by Subtraction
	T ₁₂ = 0.16 (System Penetration)	CM ₁₂ = 0.42 (Smart Cards/Other One-Time Password Tokens) LCM ₁₂ = 0.58 by Subtraction
	T ₁₃ = 0.32 (Denial of Service)	CM ₁₃ = 0.97 (Firewalls) LCM ₁₃ = 0.03 by Subtraction
V ₂ = 0.26 (External Security Breach Only)	T ₁₄ = 0.04 (Financial / Telecom Fraud)	CM ₁₄ = 0.80 (Security Audits) LCM ₁₄ = 0.20 by Subtraction
	T ₂₁ = 0.22 (Abuse of Wireless Network and Web Site Defacement)	CM ₂₁ = 0.35 (Public Key Infrastructure) LCM ₂₁ = 0.65 by Subtraction
	T ₂₂ = 0.02 (Sabotage)	CM ₂₂ = 0.35 (Intrusion Prevention Systems) LCM ₂₂ = 0.65 by Subtraction
	T ₂₃ = 0.76 (Virus)	CM ₂₃ = 0.96 (Anti -Virus Software) LCM ₂₃ = 0.04 by Subtraction
V ₃ = 0.39 (Both Internal and External Security Breaches Only)	T ₃₁ = 0.32 (Unauthorized Access to Information)	CM ₃₁ = 0.72 (Intrusion Detection Systems) LCM ₃₁ = 0.28 by Subtraction
	T ₃₂ = 0.59 (Malicious Code)	CM ₃₂ = 0.70 (Server Based Access Control) LCM ₃₂ = 0.30 by Subtraction
	T ₃₃ = 0.09 (Theft of Proprietary Information)	CM ₃₃ = 0.46 (Encrypted Files) LCM ₃₃ = 0.54 by Subtraction

for $v_1 t_1$, etc., $30 \times \$5.67 = \170.10 is calculated. The sum total, i.e., $90.5\% \times \$5.67$ per $1\% \approx \$513$ improvement cost, and $\Delta ECL = \$833.38 - \$320.00 = \$513.38$ for a lower RR are now almost identical. Fig. 7 shows how the SM is used to manage risk by improving on the countermeasure probabilities.

IX. DISCUSSIONS AND CONCLUSION

There are sufficient incentives as to why "Johnny" should and could, rather than might, evaluate security risks by making meaningful estimates [14]. While we are still having trouble obtaining accurate quantitative data, this model is at least as

Vulnerab.	Threat	CM & LCM	Res. Risk	CM & LCM	Res. Risk	Change	Cost	per 1%
0.35	0.48	0.7		1		0.3	\$170.10	\$5.67
		0.3	0.0504	0	0			
	0.16	0.42		0.42		0	\$0.00	
		0.58	0.03248	0.58	0.03248	0	\$0.00	
	0.32	0.97		0.97		0	\$0.00	
		0.03	0.00336	0.03	0.00336	0	\$0.00	
	0.04	0.8		0.8		0	\$0.00	
		0.2	0.0028	0.2	0.0028			
0.26	0.22	0.35		0.35		0	\$0.00	
		0.65	0.03718	0.65	0.03718	0	\$0.00	
	0.02	0.35		0.35		0	\$0.00	
		0.65	0.00338	0.65	0.00338			
	0.76	0.96		1		0.04	\$22.68	
		0.04	0.007904	0	0			
0.39	0.32	0.72		0.9852		0.2652	\$150.37	
		0.28	0.034944	0.0148	0.00184704			
	0.59	0.7		1		0.3	\$170.10	
		0.3	0.06903	0	0			
	0.09	0.46		0.46		0	\$0.00	
		0.54	0.018954	0.54	0.018954			
Total Risk			0.260432	Total Risk	0.10000104	0.9052	\$513.25	Total Cost
Percentage			26.04%	Percentage	10.00%			
BASE	SERVER	Final Risk	0.1041728	Final Risk	0.040000416			IMPROVED
Asset=	\$8000	ECL	\$833.38	ECL	\$320.00			SERVER
Criticality=	0.40			Delta ECL	-\$513.38			

Fig. 7. Risk management from Tables IV and VIII to break even at \$513 (difference due to round-off errors) for a 90.5% CM improvement, resulting in a final RR of 10%.

good as qualitative measures and will only provide better information to the user over time as technology advances, providing more informative data [32]. A ubiquitous use of this practical technique is the installation of the SM software with a required data bank or repository in everyone’s PC to get a daily report of your PC’s security index out of a perfect 100% to lend room for relative improvement. This way, one is informed daily about the extent of his mitigation dollars to bring the equipment to a desirable percentage of security. The difficulty in input data collection and parameter estimation for the SM model is a challenge. The presented “Center” case study satisfactorily shows that.

The author has employed the concept of simple relative frequency approach, which is otherwise known as counting techniques. Once the survey samples, or, even better, the population values with a 100% (census) if feasible, are collected, then, the vulnerability–threat–LCM risks are collected with a least sampling error. This implies that the output will approach more closely the expected theoretical value, consistently with a negligible sampling error, assuming no human (measurement) error. Then, we can quantitatively predict the RR of a system at risk. This is what the SM model aims at [15], [16]. The budgetary portfolio in terms of the ECL—at the end of the proposed quantitative analyzes—is an additional value to compare maintenance practices to assess an improvement over the conventional subjective methods [3]–[6].

Finally, the static MC or dynamic discrete-event simulation of the SM model for verifying the suggested statistical data collection proves the design’s validity. We satisfactorily get the same result: 2.69% in Figs. 5 and 6. For further research, the challenge lies on the implementation of this design model on how to classify into taxonomies records of the count of “saves and crashes” for a desired vulnerability–threat–CM track in the SM model. A simulation of cyberbreach activities can be emulated through the implementation of software projects

to mimic the expensive, risky, and compromising real-world situation. Other authors have published works on the concepts of “secure coding” [17]–[19] and vulnerability analysis [20], [21], as well as “security testing” [22] and software risk in general [23], [24]. By managing the risk as in Section VIII, the results can only be as accurate as the input data measurements studied in a server example in Section VII.

Finally, a risk management example [32] in Fig. 7 from a case study of a computer center is added to show how the SM model can effectively be employed from various surveys to illustrate how the RR can be mitigated in terms of real dollars to make sense. This is achieved by calculating a breakeven point, when the total expenses accrued for the improvement of the CM devices become identical to the positive gain in the ECL due to lowering of the RR. This practice will give the risk manager a solid base to move from. It is important to keep in mind that the security testing and surveying are integral parts of this design implementation. Therefore, for a purely quantitative risk assessment, the use of national surveys for data may be the best way to go, and “the security meter’s methodology is an excellent framework for producing statistically verified risk valuations and associated monetary costs that decision maker can find to be useful” [25]. As a recommendation for a possible extension in future research, the model could include a time value (clock) and a dynamic criticality factor. The criticality constant represents the degree of disruption to an organization caused by the total loss of the asset due to a threat and vulnerability misfortune. However, this value may increase or decrease, depending on time and date constraints. For example, the server reviewed is very critical to the center for a period of 1–3 days at the end of each month when the marketing and billing data are produced and mailed [25]. For the remaining days, the server is not nearly as critical. It could, hence, be worthwhile to attempt a plot of the risk graphically with respect to the dynamic criticality

rating over time to better assess, both monetarily and threat-wise, when the information system or asset is the most or least at risk, as well as the associated monetary impact, therefore managing the risk. This would prove useful when analyzing one's posture, for instance, before going to war, designing an upgrade, etc. We will use the same data and just change the criticality values and hit the "recomputed" button. Additionally, if the percentages for either a set of disjoint vulnerabilities, or disjoint threats for any vulnerability, do not add up to 100% (resulting in less or more than 100%) due to survey design error, then to obey the principles of the SM design, simply normalize these survey percentages by dividing each value with the probability sum. If one has 50%, 50%, 30%, and 20% summing to 150%, then the normalized percentage values are 50/1.5, 50/1.5, 30/1.5, and 20/1.5. This is why extreme care should be given to design surveys with the SM in sight.

The SM (2005) approach is not a stand-alone method [1], i.e., not supported by other authors. For instance, Landoll [30, p. 33] notes, "A quantitative approach to determining risk and even presenting security risk has the advantages of being objective and expressed in terms of dollar figures." Pandian [32, p. 238] defines a risk mitigation plan as an action plan designed to reduce risk exposure and RR as the remaining part of risk after the mitigation plan is completed. In this paper, however, the CM action is considered as a major part of the mitigation plan [32, ch. 3]. In addition, the simulations in Figs. 5 and 6 are presented to study the hypothetical scenarios when analytical methods cannot be used to conduct "what if" analyses [31], [32]. For future work, the author plans to implement "Game Theory" to optimize countermeasure actions for a best "defensive" strategy against the "hostile" threats.

ACKNOWLEDGMENT

The author thanks L. Wentzel for providing the data sets in Tables IV–VII, and also D. Tyson for JAVA programming on the security meter.

REFERENCES

- [1] M. Sahinoglu, "Security meter: A practical decision-tree model to quantify risk," *IEEE Security Privacy*, vol. 3, no. 3, pp. 18–24, May/June 2005.
- [2] E. Forni. (2002). *Certification and Accreditation*. DSD Lab., AUM Lecture Notes. [Online]. Available: <http://www.dsdlabs.com/security.htm>
- [3] B. Schneier, *Applied Cryptography*, 2nd ed. Hoboken, NJ: Wiley, 1995. Also retrieved from <http://www.counterpane.com>, 2005.
- [4] *Capabilities-Based Attack Tree Analysis*. (2005). [Online]. Available: <http://www.attacktrees.com>; www.amenaza.com
- [5] *Time to Defeat (TTD) Model*. (2005). [Online]. Available: www.blackdragonsoftware.com
- [6] D. Gollman, *Computer Security*, 2nd ed. Chichester, U.K.: Wiley, 2006.
- [7] M. Sahinoglu, *Security Meter—A Probabilistic Framework to Quantify Security Risk*, Certificate of Registration, U.S. Copyright Office, Short Form TXu 1-134-116, Dec. 2003.
- [8] M. Sahinoglu, "A quantitative risk assessment," in *Proc. Troy Business Meeting*, San Destin, FL, 2005.
- [9] M. Sahinoglu, "Security meter model—A simple probabilistic model to quantify risk," in *Proc. 55th Session Int. Stat. Inst. Conf. Abstract Book*, Sydney, Australia, 2005, p. 163.
- [10] M. Sahinoglu, "Quantitative risk assessment for software maintenance with Bayesian principles," in *Proc. ICSM*, Budapest, Hungary, 2005, vol. II, pp. 67–70.
- [11] M. Sahinoglu, "Quantitative risk assessment for dependent vulnerabilities," in *Proc. Int. Symp. Product Quality Reliab. (52nd Year)*, RAMS, Newport Beach, CA, 2006, pp. 82–85.
- [12] R. V. Hogg and A. T. Craig, *Introduction to Mathematical Statistics*, 3rd ed. New York: Macmillan, 1970. Library of Congress Catalog Card No. 74-77968.
- [13] C. Nagle and P. Cates, *CS6647—Simulation Term Project*. Montgomery, AL: Troy Univ., 2005.
- [14] G. Cybenko, "Why Johnny can't evaluate security risk," *IEEE Security Privacy*, vol. 4, no. 1, p. 5, Jan./Feb. 2006.
- [15] W. G. Cochran, *Sampling Techniques*, 3rd ed. Hoboken, NJ: Wiley, 1970.
- [16] M. Sahinoglu, D. Libby, and S. R. Das, "Measuring availability indexes with small samples for component and network reliability using the Sahinoglu–Libby probability model," *IEEE Trans. Instrum. Meas.*, vol. 54, no. 3, pp. 1283–1295, Jun. 2005.
- [17] M. Howard and D. LeBlanc, *Writing Secure Code*, 2nd ed. Redmond, WA: Microsoft, 2002.
- [18] F. Swiderski and W. Snyder, *Threat Modeling*. Redmond, WA: Microsoft, 2004.
- [19] R. Weaver, *Guide to Network Defense and Countermeasures*, 2nd ed. Washington, DC: Thomson, 2007.
- [20] O. H. Alhazmi and Y. K. Malaya, "Quantitative vulnerability assessment of systems software," in *Proc. RAMS*, Alexandria, VA, Jan. 2005, pp. 615–620.
- [21] I. Krusl, E. Spafford, and M. Tripunitara, "Computer vulnerability analysis," Dept. Comput. Sci., Purdue Univ., West Lafayette, IN, COAST TR 98-07, May 1998.
- [22] B. Potter and G. McGraw, "Software security testing," *IEEE Security Privacy*, vol. 2, no. 5, pp. 81–85, Sep./Oct. 2004.
- [23] S. A. Scherer, *Software Failure Risk*. New York: Plenum, 1992.
- [24] J. Keyes, *Software Engineering Handbook*. New York: Auerbach, 2003.
- [25] L. Wentzel, *Quantitative Risk Assessment*. Fort Lauderdale, FL: Nova Southeastern Univ., May 28, 2006. Working paper for DISS 765 (Managing Risk in Secure Systems), Spring Cluster 2008.
- [26] R. Richardson, *2005 CSIFBI Computer Crime and Security Survey*. San Francisco, CA: Comput. Security Inst., 2005. Retrieved from <http://www.goosi.com> on May 4, 2006.
- [27] S. Berinato, *The Global State of Information Security 2005*, 2005. Retrieved from <http://www.cio.com/archive/091505/global.html> on May 15, 2006.
- [28] A. Melek and M. MacKinnon, *2005 Global Security Survey*, May 13, 2006, Deloitte Touche Tohmatsu. [Online]. Available: <http://www.deloitte.com/dtt/cda/doc/content/Deloitte2005GlobalSecuritySurvey.pdf>
- [29] NASA, *NASA Procedural Requirements, Physical Security Vulnerability Risk Assessments*, 2004. Document NPR 1620.2, expires Jul. 15, 2009.
- [30] D. Landoll, *The Security Risk Assessment Handbook*. Boca Raton, FL: Auerbach, 2006.
- [31] C. R. Pandian, *Applied Software Risk Management—A Guide for Software Project Managers*. Boca Raton, FL: Auerbach, 2007.
- [32] M. Sahinoglu, *Trustworthy Computing—Analytical and Quantitative Engineering Evaluation*. Hoboken, NJ: Wiley, Aug. 2007.



Mehmet Sahinoglu (S'78–M'81–SM'93) received the B.S. degree in electrical engineering from Middle East Technical University, Ankara, Turkey, the M.S. degree in electrical engineering from the University of Manchester Institute of Science and Technology, Manchester, U.K., and the Ph.D. degree in electrical and computer engineering and statistics from Texas A&M University, College Station.

He is currently with the Department of Computer Science, Troy University, Montgomery, AL. He originated the Sahinoglu–Libby (SL) pdf, jointly with D. Libby, in 1981 (see [16]). He is the author of "Compound Poisson software reliability model" (*IEEE Trans. Software Eng.*, Jul. 1992) and "Compound Poisson stopping rule algorithm" (*Wiley J. Testing, Verification, Rel.*, Mar. 1997), which is about cost-effective software testing and, recently, "The security meter" (*IEEE Security and Privacy*, Apr./May 2005), which deals with quantifying risk. He is the author of *Trustworthy Computing* (Wiley, 2007), with a CD ROM, which is a book on security and reliability.

Dr. Sahinoglu is a Fellow of the Society of Design and Process Science, a Member of the Armed Forces Communications and Electronics Association and the American Statistical Association, and an Elected member of the International Statistical Institute and the International Association of Statistical Computing. He is a 2006 Microsoft Research Scholar on the Trustworthy Computing curriculum—one of the 14 awardees around the globe.