QUALITY AND RELIABILITY ENGINEERING INTERNATIONAL
Qual. Reliab. Engng. Int. 2001; **17**: 105–112

AN ANOMALY DETECTION TECHNIQUE BASED ON A CHI-SQUARE STATISTIC FOR DETECTING INTRUSIONS INTO INFORMATION SYSTEMS

NONG YE*AND QIANG CHEN
Department of Industrial Engineering, Arizona State University, Tempe, AZ 85287-5906, USA

SUMMARY

An intrusion into an information system compromises its security (e.g. availability, integrity and confidentiality) through a series of events in the information system. Intrusive events often show departures (anomalies) from normal events in an information system. This paper presents an anomaly detection technique based on a chi-square statistic. This technique builds a profile of normal events in an information system—a norm profile computes the departure of events in the recent past from the norm profile and detects a large departure as an anomaly—a likely intrusion. This technique was tested for its performance in distinguishing normal events from intrusive events in an information system. The test results demonstrated the promising performance of this technique for intrusion detection in terms of a low false alarm rate and a high detection rate. Intrusive events were detected at a very early stage. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: computer security; intrusion detection; multivariate analysis; chi-square statistic

INTRODUCTION

An intrusion into an information system compromises its security (e.g. availability, integrity and confidentiality) through a series of events in the information system [1–4]. For example, a denial-of-service intrusion compromises the availability of an information system by flooding a constituent server with an overwhelming number of service requests to the server over a short period of time and thus denies or degrades the service to legitimate users. Another intrusion may compromise the integrity and confidentiality of an information system by gaining root privileges and then modifying and stealing information.

As we increasingly rely on information systems to support critical operations in defense, banking, telecommunication, transportation, electric power and many other systems, intrusions have become a significant threat to our society with potentially severe consequences. Layers of defense can be set up against intrusions through prevention, detection, reaction and so on. Some examples of intrusion prevention techniques are firewalls and guards, authentication, and encryption. Intrusion detection identifies intrusions being leaked through the fence of prevention and acting on an information system.

This paper presents an intrusion detection technique based on a chi-square test statistic. The rationale for developing this technique is first presented after reviewing existing intrusion detection techniques. Then the technique is described and an intrusion detection application is defined. Finally the results of testing the technique are presented and discussed.

REVIEW OF EXISTING INTRUSION DETECTION TECHNIQUES

Existing intrusion detection techniques fall in two major categories: signature recognition and anomaly detection [5–12]. Signature recognition techniques [4,5,9] store the signatures of known intrusion scenarios, match the observed behavior with these intrusion signatures and signal an intrusion when there is a match. Signature recognition techniques have a limitation in that they cannot detect novel attacks whose signatures are unknown.

The limitation of signature recognition techniques can be overcome by using anomaly detection techniques as a complement. For a subject of

*Correspondence to: Nong Ye, Department of Industrial Engineering, Arizona State University, Box 875906, Tempe, AZ 85287-5906, USA. Email: nongye@asu.edu
Contract/grant sponsor: Air Force Office of Scientific Research
Contract/grant sponsor: Air Force Research Laboratory
Contract/grant sponsor: Defense Advanced Research Projects Agency

Received 26 June 2000
Revised 9 January 2001

interest (user, file, privileged program, host machine, computer network etc.) anomaly detection techniques [6–12] establish a profile of the subject's normal behavior (norm profile), compare the observed behavior of the subject with its norm profile, and signal an intrusion when the subject's observed behavior departs from its norm profile. Hence, anomaly detection techniques can detect both known and novel intrusions if they demonstrate departures from a norm profile.

Intrusive behavior often shows departures (anomalies) from normal behavior in an information system. For example, in a denial-of-service intrusion through flooding a server, the intensity of events to the server is much higher than the event intensity in a normal operation condition. In an intrusion through gaining root privileges, actions that an intruder takes to get into the information system and maneuver inside the information system are often different from actions of legitimate users in a normal operation condition. Hence, anomalies can be used to detect possible intrusions.

In the existing anomaly detection techniques, a norm profile is specified in one of the following forms: strings [7], formal logic [8], production rules [9] and statistical distributions [10–12]. Forrest *et al.* [7] simulate the human immune system and model the norm profile of a subject as a set of binary strings. For a set of normal strings (self strings), a set of detector strings is constructed so that detector strings do not match self strings. If an incoming string matches any of the detector strings for at least the r number of contiguous bits, the detection of an anomaly (a foreign object or non-self) is declared. The string-based anomaly detection technique has been applied to detecting anomalous sequences of system calls to the kernel of a UNIX system. However, this technique has several drawbacks. First, there may exist some non-self strings for which it is impossible to generate detector strings. This issue still remains to be resolved [7]. Second, this technique is not robust to noises. Noises in a self string could lead to a match between the self string and one of the detector strings, causing a false alarm. Noises in a non-self string could make the string not matched to any of the detector strings, causing a miss. The sensitivity to noises depends heavily on the parameter r. On one hand, a small r increases the rate of false alarms. On the other hand, a large r increases the rate of misses. Currently, the parameter r is determined empirically.

The logic-based anomaly detection technique [8] has been applied to routers, Domain Name System and some privileged programs. However, formal logic is difficult for most system administrators to understand and use for specifying a norm profile. In contrast, production rules in expert systems [9] are more natural and understandable than formal logic for most system administrators to specify and update a norm profile.

Both logic- and rule-based anomaly detection techniques have a limitation. It is difficult to enumerate and specify all possibilities of normal behavior, especially when multiple subjects (e.g. objects and actions in an information system) are involved. Moreover, the behavior of a subject such as a user is generally not fixed but dynamically changing. It is difficult to specify the dynamically changing behavior in advance using formal logic or production rules.

In the IDES/NIDES systems [10–12], a statistical-based anomaly detection technique is used to represent the expected normal behavior of a subject and variance due to noises. The statistical-based anomaly detection technique overcomes the problems with the string-based, logic-based and rule-based anomaly detection technique in handling noises and variances. However, the statistical technique in IDES/NIDES is a univariate technique that is applied to only one behavior measure, whereas many intrusions involve multiple subjects and multiple actions having impact on multiple behavior measures. Hence, a multivariate anomaly detection technique is needed for intrusion detection.

A MULTIVARIATE STATISTICAL TECHNIQUE

Multivariate process control techniques [13] such as Hotelling's T^2 [14], multivariate cumulative sum (MCUSUM) [15] and multivariate exponentially weighted moving average (MEWMA) [16] are typically used to monitor and detecting anomalies of a process in a manufacturing system. Theoretically, these multivariate process control techniques can be applied to intrusion detection for monitoring and detecting anomalies of a process in an information system. Practically, the computationally intensive procedure of these multivariate process control techniques cannot meet the demands of intrusion detection for several reasons. First, intrusion detection must deal with large volumes of high-dimensional process data due to a large number (e.g. hundreds or thousands) of behavior measures and a high frequency of event occurrence. Second, intrusion detection requires a minimum delay of processing each event in an information system to ensure an early indication and warning of intrusions.

For example, let us consider the computational procedure of Hotelling's T^2. Let $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})'$ denote an observation of p mea-

sures from a process at time i. Assume that when the process is operating normally (in control), the population of $\boldsymbol{X}$ follows a multivariate normal distribution with the mean vector $\boldsymbol{\mu}$ and the covariance matrix Σ. Using a data sample of size n, the sample mean vector $\overline{\boldsymbol{X}}$ and the sample covariance matrix $\boldsymbol{S}$ are usually used to estimate μ and Σ [13], where

$$\overline{\boldsymbol{X}} = (\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_p) \tag{1}$$

$$\boldsymbol{S} = \frac{1}{n-1}\sum_{i=1}^{n}(\boldsymbol{X}_i - \overline{\boldsymbol{X}})(\boldsymbol{X}_i - \overline{\boldsymbol{X}})'. \tag{2}$$

Hotelling's T^2 statistic for an observation $\boldsymbol{X}_i$ is determined by the following [13]:

$$T^2 = (\boldsymbol{X}_i - \overline{\boldsymbol{X}})'\boldsymbol{S}^{-1}(\boldsymbol{X}_i - \overline{\boldsymbol{X}}). \tag{3}$$

A large computed value of T^2 indicates a large deviation of the observation $\boldsymbol{X}_i$ from the in-control population. We can obtain a transformed value of the T^2 statistic, $n(n-p)T^2/(p(n+1)(n-1))$ which follows an F distribution with p and $(n-p)$ degrees of freedom, by multiplying T^2 by the constant $n(n-p)/(p(n+1)(n-1))$. If the transformed value of the T^2 statistic is greater than the tabulated F value for a given level of significance, α, then we reject the null hypothesis that the process is in control (normal) and thus signal that the process is out of control (anomalous).

The computation in equation (3) involves a covariance matrix and its inverse. Even modern computers have difficulty in storing a large covariance matrix for hundreds or thousands of variables in the memory. Moreover, there may be hundreds or thousands of events occurring in an information system within a short period of time. If we compute the Hotelling's T^2 statistic for all events that occur at a high frequency, the computation time becomes unbearable, and the processing delay of computing the covariance matrix and its inverse for each event becomes unacceptable.

Therefore, a multivariate anomaly detection technique with a low computation cost is needed for intrusion detection. Since the Hotelling's T^2 statistic is a measure of the statistical distance from an observation to the mean estimate of the multivariate normal distribution, we develop a distance measure based on a chi-square test statistic as follows [17]:

$$X^2 = \sum_{i=1}^{n}\frac{(X_i - E_i)^2}{E_i} \tag{4}$$

where X_i is the observed value of the ith variable, E_i is the expected value of the ith variable and n is the number of variables. X^2 is small if an observation of the variables is close to the expectation. Using $(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_n)$ as the estimate of the expectation, we obtain:

$$X^2 = \sum_{i=1}^{n}\frac{(X_i - \overline{X}_i)^2}{\overline{X}_i} \tag{5}$$

According to the central limit theorem, when the number of variables is large enough (i.e. greater than 30), X^2 as the sum of squared differences between the observed and expected values of those variables has approximately a normal distribution. Hence, the interval of $[\mu - Z_{\alpha/2}\sigma, \mu + Z_{\alpha/2}\sigma]$ contains $(1-\alpha)$ per cent of the possible values of the X^2 population, where μ and σ are the mean and variance of the X^2 population, α is the significance level and $Z_{\alpha/2}$ is the tabulated value of the standard normal distribution. The mean and standard deviation of the X^2 population can be estimated from the sample data of X^2 by the sample mean $\overline{X^2}$ and the sample standard deviation S_X^2. The in-control limits to detect anomalies can be set to 3 to obtain 3-sigma control limits [18,19], $[\overline{X^2} - 3S_X^2, \overline{X^2} + 3S_X^2]$. Since we are interested in detecting significantly large X^2 values (large differences between observed and expected values) for intrusion detection, we need to set only the upper control limit to $\overline{X^2} + 3S_X^2$. That is, if the computed X^2 for an observation is greater than $\overline{X^2} + 3S_X^2$, we signal an anomaly.

Unlike the Hotelling's T^2 statistic that is a distance measure taking into account the covariance of multiple variables, the X^2 statistic in equation (5) does not consider the relationships of multiple variables in order to simplify the computation. If intrusions into information systems cause only small violations of variable relationships but large departures from the mean in some of multiple variables, the X^2 statistic can be as effective as the Hotelling's T^2 statistic for intrusion detection.

AN APPLICATION

This section describes an intrusion detection application, including the data source, training data, testing data and representation of the application problem.

Data source

An information system typically consists of host machines (e.g. machines running a UNIX operation system and machines running the Windows NT operating system) and communication links

connecting those host machines, forming a network of host machines. Two sources of data have been widely used to capture events in an information system for intrusion detection: network traffic data and audit trail data (audit data). Network traffic data contain data packets traveling over communication links between host machines to capture events over communication links. Audit data capture events occurring on a host machine. In this study, we use audit data from a UNIX-based host machine (specifically a Sun SPARC 10 workstation with the Solaris operating system) and focus on intrusions into a host machine that leave trails in audit data.

The Solaris operating system from Sun Microsystems Inc. has a security facility, called the Basic Security Module (BSM). BSM supports the monitoring of activities in a host machine by recording security-relevant events. There are over 250 different types of BSM auditable events, depending on the version of the Solaris operating system. Since there are about 284 different types of BSM audit events on our host machine, we consider 284 event types in this study. A BSM audit record for each event contains a variety of information, including the event type, user ID, group ID, process ID, session ID, the system object accessed and so on. In this study, we extract and use only the event type that was one of the most critical characteristics of an audit event.

Training and testing data

For intrusion detection, we want to build a long-term profile of normal events and to compare events in the recent past to the long-term norm profile for detecting a significant departure. Audit data of normal events are required for training the norm profile. In this study, we use a sample of audit data for normal events that is developed by the MIT (Massachusetts Institute of Technology) Lincoln Laboratory. The sample contains a stream of 3019 audit events. We use the first part of the audit data, consisting of 1613 audit events, for training a norm profile. The second part of the audit data, consisting of 1406 audit events, is used for testing.

We also need the audit data of intrusive events for testing. To create the audit data of intrusive events, we use a number of intrusion scenarios that we have collected over years from various information sources. Some examples of the intrusion scenarios are the password guessing, use of symbolic links to gain root privileges, attempts to gain an unauthorized remote access etc. We run these intrusion scenarios in a random order through separate sessions on our host machine to obtain the audit data of these intrusion sessions. The audit data of those intrusion sessions, containing 1225 audit events, are used for testing. Hence, the testing data contain 1406 audit events for normal events and 1225 audit events for intrusive events.

Problem representation

Activities on a host machine are captured through a continuous stream of audit events, each of which is characterized by the event type. For intrusion detection, we want to build a long-term profile of normal events, and to compare events in the recent past to the long-term norm profile for detecting a significant departure. We define events in the recent past from time $t - k$ to time t by a vector of $(X_1, X_2, \ldots, X_{284})$ for 284 event types respectively, based on the exponentially weighted moving average technique [18,19]. At time t, the audit events in the recent past from time $t - k$ to time t are summarized as follows.

$$\begin{aligned} X_i(t) &= \lambda * 1 + (1 - \lambda) * X_i(t-1) \\ &\quad \text{if the audit event at time } t \text{ falls into the } i\text{th} \\ &\quad \text{event type} \\ X_i(t) &= \lambda * 0 + (1 - \lambda) * X_i(t-1) \\ &\quad \text{if the audit event at time } t \text{ is different} \\ &\quad \text{from the } i\text{th event type} \end{aligned} \tag{6}$$

where $X_i(t)$ is the observed value of the ith variable in the vector of observation at time t, λ is a smoothing constant that determines k or the decay rate, and $i = 1, \ldots, 284$. The most recent observation at time t receives a weight of λ, the observation at time $t - 1$ receives a weight of $\lambda(1 - \lambda)$, and the observation at time $t - k$ receives a weight of $\lambda(1 - \lambda)^k$.

In this study, we initialize $X_i(0)$ to 0 for $i = 1, \ldots, 284$. We let λ be 0.3 which is a typical value for the smoothing constant. Figure 1 shows the decay effect of the smoothing constant 0.3.

Hence, for each audit event in the training and testing data, we obtain a vector of $(X_1, \ldots, X_{284})$. For example, given the following stream of audit events:

$t = 0,$	1,	2,	3,	...
	Event Type 3,	Event Type 8,	Event Type 1,	...

At $t = 0$, all variables in the vector of $(X_1, \ldots, X_{284})$ have a value of 0. At time $t = 1$, X_3 has a value of 0.3 (equal to $0.3 * 1 + 0.7 * 0$) and all other variables have a value of 0. At time $t = 2$, X_3 has a value of 0.21 (equal to $0.3 * 0 + 0.7 * 0.3$), X_8 has a value of 0.3

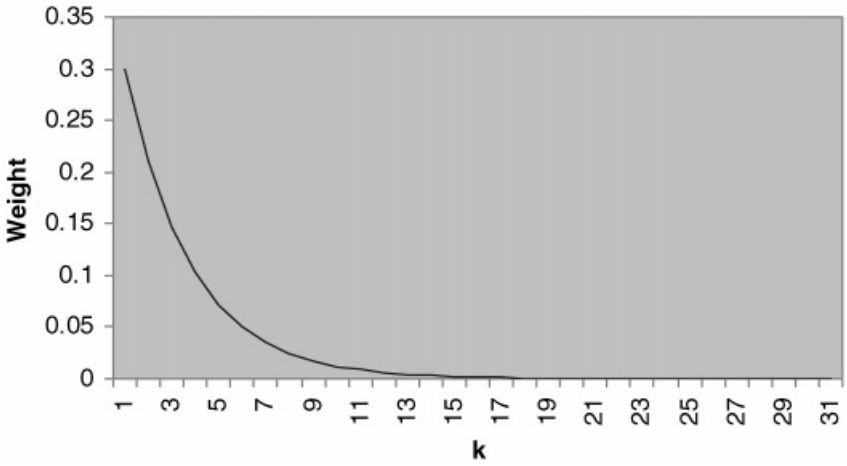

Figure 1. The effect of the smoothing constant 0.3

(equal to $0.3 * 1 + 0.7 * 0$) and all other variables have a value of 0. At $t = 3$, X_3 has a value of 0.147 (equal to $0.3 * 0 + 0.7 * 0.21$), X_8 has a value of 0.21 (equal to $0.3 * 0 + 0.7 * 0.3$), X_1 has a value of 0.3 (equal to $0.3 * 1 + 0.7 * 0$) and all other variables have a value of 0.

The expected vector of $(X_1, \ldots, X_{284})$ for the long-term profile of normal activities is estimated from the training data by averaging all 1613 vectors that we obtain from 1613 audit events in the training data, to obtain the expected vector $(\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_{284})$. Considering that events in a host machine actually arrive not all at once but sequentially, we use the following recursive formula to incrementally update $\overline{X}_i$ after each event:

$$\overline{X_{k,i}} = \frac{(k-1)\overline{X_{k-1,i}} + X_{k,i}}{k} \tag{7}$$

where k is the index of the current event.

It is possible that some types of audit events do not appear in the training data but occur in the testing data. Hence, the average of the variable for such an event type is zero after the training. To avoid having a zero value for the denominator of equation (5), the average of the variable for such an event type is assigned to a small value after the training. In this study, we used 0.00001.

For each of the audit events (1406 audit events for normal activities and 1223 audit events for attack activities) in the testing data and the corresponding observed vector of $(X_1, \ldots, X_{284})$, we compute X^2 as follows:

$$X^2 = \sum_{i=1}^{284} \frac{(X_i - \overline{X_i})^2}{\overline{X_i}} \tag{8}$$

The computed X^2 is small if the observed vector is close to the expected vector, in other words the observation is close to the norm profile.

To determine the upper limit of X^2 in terms of $\overline{X^2} + 3S_X^2$, we use the computed X^2 values for the first half (first 703 audit events) of 1406 normal events in the testing data to estimate the average $(\overline{X^2})$ and the standard deviation (S_X^2). The upper limit, $\overline{X^2} + 3S_X^2$, is then used to determine whether we should signal each of the remaining 703 normal events and 1225 intrusive events in the testing data as an anomaly. If the computed X^2 for an audit event is greater than the upper limit, we signal the audit event as an anomaly.

RESULTS AND DISCUSSIONS

Using the computed X^2 values for the first 703 normal events in the testing data, we obtain 1.72, 1.71 and 6.85 for $\overline{X^2}$, S_X^2, and $\overline{X^2} + 3S_X^2$ respectively. That is, the upper limit of the computed X^2 values for normal events is 6.85. If a computed X^2 value for an audit event is greater than 6.85, we signal this audit event as an anomaly.

Figure 2 shows the computed X^2 values for the remaining 703 normal events (event no. 1–703) and 1225 intrusive events (event no. 704–1928) in the testing data. Table 1 summarizes the statistics (minimum, maximum, average and standard deviation) of the computed X^2 values for the 703 normal events and the 1225 intrusive events. As shown in Figure 2 and the statistics in Table 1, the computed X^2 values of the normal events are on average smaller than the computed X^2 values of the intrusive events. Note that

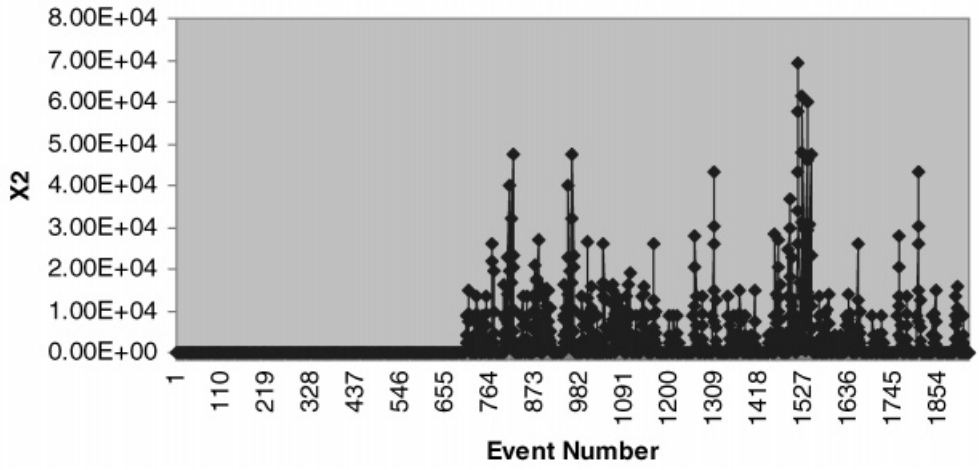

Figure 2. The computed X_2 values of audit events in the testing data

Table 1. The statistics of the computed X_2 values for normal and attack data in the testing

Testing Data	Minimum	Maximum	Average	Standard Deviation
Normal data	5.29E−01	4.06E+00	1.56E+00	9.23E−01
Attack data	6.81E−01	6.92E+04	3.79E+03	7.77E+03

the smaller the computer X^2 is, the closer the audit event is to the norm profile.

When we use the upper limit of 6.85 to examine the computed X^2 values for the 703 normal events, we obtain no signals. This indicates that there are no false alarms, in other words, we obtain the 0% false alarm rate. A false alarm is a signal for a normal event.

When we use the upper limit of 6.85 to examine the computed X^2 values for the 1225 intrusive events, we obtain 873 signals. The detection rate by audit event is 75% (equal to 873/1225) for the 1225 intrusive events. Since an intrusion session corresponds to one intrusion scenario, we group the 1225 intrusive events into individual intrusion sessions. As a result, there are signals for each individual intrusion session. The minimum detection rate by audit event for all the intrusion sessions is 35% and the maximum detection rate by audit event for all the intrusion sessions is 86%. Hence, there are signals for each intrusion session. The detection rate by intrusion session is 100%. 71% of the intrusion sessions are detected at the first audit event. That is, the first audit event of these intrusion sessions is signaled. The remaining 29% of the intrusion sessions are detected at the second audit event. Hence, every intrusion session is detected at a very early stage.

The 75% detection rate by audit event for intrusions is most likely attributed to the fact that intrusions involve some behavior (e.g. commands) that is also common in normal activities. Hence, we should not expect that every audit event in an intrusion session be detected. However, the overall behavior of intrusions seems different statistically from the overall behavior of normal activities, as we observe in this study from the difference between 0% detection rate by audit event for normal activities and 75% detection rate by audit event for intrusions. It is also possible that some behavior in intrusions may occasionally occur in normal activities, causing false alarms, although such a phenomenon does not show up in the testing data of this study.

Therefore, we expect that more accurate detection results can be achieved at an aggregate behavior level where dominant behavior emerges and persists despite noises, rather than at the level of individual audit events. An aggregate behavior level can be the level of a session as in this study. For example, in this study we calculate the detection rate by session. Since the maximum detection rate for normal sessions is 0% and the minimum detection rate for intrusion sessions is 35%, we can select any detection rate by session in the range of (0%, 35%) as the decision threshold (e.g. 20%). Using this decision threshold, we are able to clearly distinguish intrusion sessions from normal

sessions with the 100% detection rate and the 0% false alarm rate.

On the other hand, it is also desirable to detect an intrusion session early on while it is in progress rather than waiting until the end of the session to get the detection rate by session. A balance between accurate detection and early detection can be achieved by setting three levels of intrusion detection: the normal level (green light), the alert level (yellow light) and the alarm level (red light). When we receive the first signal in a session, we trigger an alert on this session and change the color of the detection light from green to yellow, calling for the attention of the system administrator. We calculate the accumulated detection rate from the first audit event to the current audit event for the session. When the accumulated detection rate up to the current audit event of the session exceeds a decision threshold, we trigger an alarm and change the color of the detection light from yellow to red, claiming the session as an intrusion. Hence, an alert does not become an alarm until the accumulated detection rate up to the current audit event of a session exceeds a decision threshold.

The promising performance of the X^2 statistic as a simplified distance measure for intrusion detection indicates that intrusions may not manifest as violations of variable relationships but large departures from the mean in some variables. A report [20] on an application of the Hotelling's T^2 statistic to intrusion detection reveals that the performance of the Hotelling's T^2 statistic is similar to the performance of the X^2 statistic in this study.

In summary, the results of this study show that the multivariate statistical technique based on the chi-square test statistic achieved the 0% false alarm rate and the 100% detection rate by session. All intrusion sessions are detected at the first or second audit event. Although in this study the multivariate statistical technique is tested using a small set of the testing data, the results of this study demonstrate the very promising potential of this technique for intrusion detection. This technique will be tested in the future with a large set of testing data for further evaluating the performance and scalability.

ACKNOWLEDGEMENTS

This work is sponsored in part by the Air Force Research Laboratory—Rome (AFRL-Rome) under agreement number F30602-98-2-0005, the Air Force Office of Scientific Research (AFOSR) under grant number F49620-99-1-0014, and the Defense Advanced Research Projects Agency (DARPA) under grant number F30602-99-1-0506. The U.S. government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of AFRL-Rome, AFOSR, DARPA, or the U.S. Government.

REFERENCES

1. Stallings M. *Network and Inter-network Security Principles and Practice*. Prentice Hall: Englewood Cliffs, NJ, 1995.
2. Kaufman C, Perlman R, Speciner M. *Network Security: Private Communication in a Public World*. Prentice Hall: Englewood Cliffs, NJ, 1995.
3. Ye N, Giordano J, Feldman J. Detecting information warfare attacks: Current state of the art from a process control viewpoint. *Communications of the ACM*, in press.
4. Escamilla T. *Intrusion Detection: Network Security Beyond the Firewall*. John Wiley & Sons: New York, 1998.
5. Lippmann R, Fried D, Graf I, Haines J, Kendall K, McClung D, Weber D, Webster S, Wyschogrod D, Cunningham R, Zissman M. Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. *Proceedings of the DARPA Information Survivability Conference and Exposition*, January 2000. IEEE Computer Society: Los Alamitos, CA, 2000; 12–26.
6. Ghosh AK, Schwatzbard A, Shatz M. Learning program behavior profiles for intrusion detection. *Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, April 1999. Santa Clara, CA. http://www.rstcorp.com/~anup/.
7. Forrest S, Hofmeyr SA, Somayaji A. Computer immunology. *Communications of the ACM* 1997; **40**(10):88–96.
8. Ko C, Fink G, Levitt K. Execution monitoring of security-critical programs in distributed systems: A specification-based approach. *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, 1997; 134–144.
9. Anderson D, Frivold T, Valdes A. Next-generation intrusion detection expert system (NIDES): A summary. *Technical Report SRI-CSL-97-07*, SRI International, Menlo Park, CA, May 1995.
10. Javitz HS, Valdes A. The SRI statistical anomaly detector. *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*, May 1991.
11. Javitz HS, Valdes A. The NIDES statistical component description of justification. *Technical Report A010*, SRI International, Menlo Park, CA, March 1994.
12. Jou Y, Gong F, Sargor C, Wu X, Wu S, Chang H, Wang F. Design and implementation of a scalable intrusion detection system for the protection of network infrastructure. *Proceedings of the DARPA Information Survivability Conference and Exposition*. IEEE Computer Society: Los Alamitos, CA, 2000; 69–83.
13. Ryan TP. *Statistical Methods for Quality Improvement*. John Wiley & Sons: New York, 2000.
14. Hotelling H. Multivariate quality control. *Techniques of Statistical Analysis*. Eisenhart C, Hastay MW, Wallis WA (eds.). McGraw-Hill: New York, 1947.
15. Woodall WH, Ncube MM. Multivariate CUSUM quality-control procedures. *Technometrics* 1985; **27**:185–192.
16. Lowry CA, Woodall WH, Champ CW, Rigdon SE. Multivariate exponentially weighted moving average control chart. *Technometrics* 1992; **34**:46–53.

17. Daniel WW. *Biostatistics: A Foundation for Analysis in the Health Sciences*. John Wiley & Sons: New York, 1987.
18. Montgomery DC. *Introduction to Statistical Quality Control*. John Wiley & Sons: New York, 2000.
19. Banks J. *Principles of Quality Control*. John Wiley & Sons: New York, 1989.
20. Ye N, Chen Q, Emran SM, Vilbert SM. A multivariate quality control technique for cyber intrusion detection. *IEEE Transactions on Computers* 2001; in review.

Authors' biographies:

Nong Ye is an Associate Professor of Industrial Engineering at Arizona State University (ASU). She holds a PhD from Purdue University, West Lafayette, Indiana. She is the Director of the Information and Systems Assurance Laboratory at ASU. Her research work focuses on assuring process quality and reliability of information systems, manufacturing and enterprise systems, and human-machine systems. She has published more than 70 journal and conference articles. She is a senior member of the Institute of Industrial Engineers, and a member of IEEE. She is a member of the Editorial Boards of the International Journal of Human-Computer Interaction and the International Journal of Cognitive Ergonomics.

Qiang Chen is a doctoral student in the Department of Industrial Engineering, Arizona State University.

