A General Risk Assessment of Security in Pervasive Computing

Yong Chen[1], Christian Damsgaard Jensen[2], Elizabeth Gray[1], Vinny Cahill[1], Jean-Marc Seigneur[1]

[1] Distributed Systems Group, Department of Computer Science, Trinity College of Ireland
Email: cheny@tcd.ie
[2] Informatics and Mathematical Modeling, Technical University of Denmark

Abstract-- There is currently much research into using trust-based mechanisms to secure computing in ubiquitous environments, which are typified by unforeseen circumstances, unexpected interactions, and unknown entities. Risk evaluation becomes essential to the trust-based decision-making process in these security mechanisms, especially when the trustworthiness of some entity is unknown and no recommendation information is available. However, risk probability estimating remains unsolved for most application scenarios.

In this paper, to measure the risk associated with some interaction, a risk probability assessment formula is presented and the traditional methods in actuarial and insurance industry are investigated. Furthermore, we propose a general risk probability definition by which a clustering procedure with Mahalanobis distance as similarity measure is presented to solve to this kind of problem. An experiment on intrusion detection system is provided to demonstrate it is feasibility and suitability for use within a trust-based security mechanism.

Index Terms-- **Security, Trust, Risk Probability, Cluster, Mahalanobis distance**

A. Introduction

A global ubiquitous computing infrastructure is envisioned in which billions of autonomous entities must interact in a decentralized and ad hoc manner. In this type of environment, traditional security mechanisms based on a centralized authorization model will not scale.

The interactions between these autonomous entities are similar to those interactions in human networks. Humans must often make ad hoc decisions regarding interaction with partially-known or unknown persons in situations where complete information is unavailable and where no trusted third party exists. Similarly, entities in the ubiquitous computing environment are both autonomous and mobile, and must be capable of dealing with unforeseen circumstances ranging from unexpected interactions with other unknown entities to disconnected operation. Human society has developed the concept of trust to overcome initial suspicion and gradually evolve privileges in these scenarios. Recently, there has been an increased interest in the development of security mechanisms for this type of environment based on the human notion of trust [1, 2, 3].

We must characterize the extent to which risk is associated with a privilege that may be assigned to an unknown entity such that the entity's trustworthiness may then be used to decide whether the risk is acceptable in granting said privilege. Risk is the possibility of something adverse happening, and risk management is the process of assessing risk, taking steps to reduce risk to an acceptable level and maintaining that level of risk. In her paper [4], Dr. Sharon Fletcher asserts that risk management has gone through two generations already, and that it needs to enter its third. Until now, there have been quite a few tools and methods proposed, but most of them still view risk assessment as a fairly static procedure. No current work provides an acceptable solution to risk assessment in the ubiquitous computing environment.

The SECURE research project focuses on the integration of trust and risk in making security decisions in the pervasive computing environment [5]. As illustrated in the SECURE framework diagram presented in Figure 1, risk evaluation is fundamental component in performing trust-based access control.

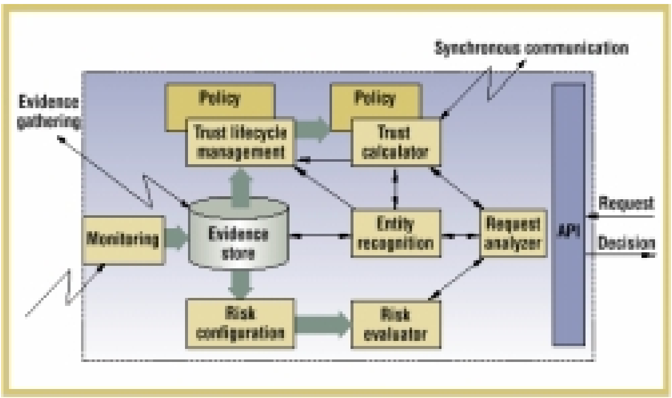


Figure 1: SECURE Framework [5]

Risk is commonly defined as the hazard level combined with the likelihood of the hazard leading to an accident and the hazard exposure or duration (latency) [6]. In this paper, we address estimating risk probability for a certain interaction, i.e., the likelihood of the hazard leading to an accident. The rest of the article is structured as follows. Section 2 addresses a risk probability assessment model to cover all possible outcomes of an interaction. Section 3 investigates risk probability estimating in actuarial and insurance industry. By reviewing a simple example, we extend the underlying idea and present a general risk probability definition. Section 4 describes our algorithm based on clustering. Mahalanobis is used to measure the similarity between any two interactions precisely and take correlation of the different features into account. In Section 5, the experimental results of applying the algorithm to intrusion detection are given. Finally, section 6 concludes and presents future work.

B. Risk Probability Estimation Model

We assume that any interaction in ubiquitous computing environments can be expressed as a feature vector and that the vector elements are comparable. This is reasonable as different discrete numbers can denote even different non-numeric features. The features must be able to describe an interaction precisely and completely. Therefore, the features describing the context of and the principals engaged in an interaction must be specified as precisely as possible.

To measure the risk associated with an interaction, the following general risk assessment formula is presented.

$$R = F\left(x_1, x_2, x_3, \ldots x_m\right) + Z$$

or

$$R = F\left(X\right) + Z$$

Where R is the probability of risk of a certain interaction, or indeed the risk value itself. X is the feature vector, and $x_i \left(i = 1,2,3 \ldots m\right)$ are its elements which consist of known parameters for this interaction. The feature elements specify the context of the interaction, participants and relevant historical memory. Their values are derived from observation or collected data. Z is the random disturbance factor, and normally we assume it to be zero. F maps the current context and participant features to the risk value or risk probability. Its specific format may be known or unknown depending on different contexts.

When the map F is known, which might be a linear or non-linear function, estimating R is not difficult. For example, consider the probable risk of vehicular death by driver's age. If it is a normal distribution, F has a format like $\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-u}{\sigma}\right)}$. According to sample data, the parameters u, σ are easy to confirm. These are the traditional parameters used in an estimating procedure.

If the dimension of vector X is high, up to 3, estimating R is also easy as we can plot out the data and see the pattern underlying the huge data set.

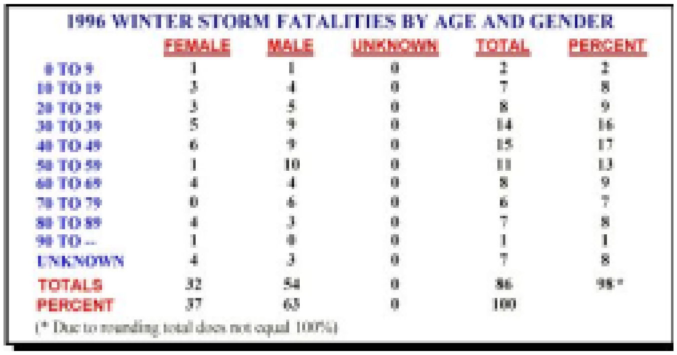
However, in fact, the map F is often unknown and the dimension of the features vector is very high. For this kind of situation, F is just like a black box passing the current data and historical data as inputs and outputting a risk value. It is worthwhile to note that the historical data is necessary for estimating the risk of the current interaction. For example, it is difficult to imagine that a child could give a risk estimation for driving, as he would have not had enough experience in this context, i.e. historical data. However, even for the historical interactions, we know only if the result of each interaction is unexpected or not. The risk probability associated with the interactions remains unknown. If the historical features vector with confirmed risk probability were known to us, there would be many means to find the map F and predict the probability of risk for a new interaction. The non-linear regression, regression splines and neural network provide the mathematical solutions to it.

In the next section we will present a general risk probability definition aimed to solve this problem.

C. Risk Probability Definition

In order to find the risk probability for each interaction, let us examine a quite simple example of risk estimation in the insurance industry. Traditionally, insurance uses historical claims data to model the probability density function (PDF) of risk. So it is necessary to review how to produce a PDF for a case.

Consider the aforementioned Winter Storm Fatalities for 1996 in America [7]. Table 1 records the sample results.



1996 WINTER STORM FATALITIES BY AGE AND GENDER

	FEMALE	MALE	UNKNOWN	TOTAL	PERCENT
0 TO 9	1	1	0	2	2
10 TO 19	3	4	0	7	8
20 TO 29	3	5	0	8	9
30 TO 39	5	9	0	14	16
40 TO 49	6	9	0	15	17
50 TO 59	1	10	0	11	13
60 TO 69	4	4	0	8	9
70 TO 79	0	6	0	6	7
80 TO 89	4	3	0	7	8
90 TO --	1	0	0	1	1
UNKNOWN	4	3	0	7	8
TOTALS	32	54	0	86	98 *
PERCENT	37	63	0	100	

(* Due to rounding total does not equal 100%)

Table 1**:** Winter Storm Fatalities for 1996 [7]

Except the unknown data, by plotting the fatalities percentage in y- axes, age in x- axes; we could see the curve is very similar to the normal distribution curve (Figure 2). A concrete PDF function can be calculated by estimating the parameters. Then we can use the age of a new principal as the input to the PDF function to output the risk of winter storm fatality.

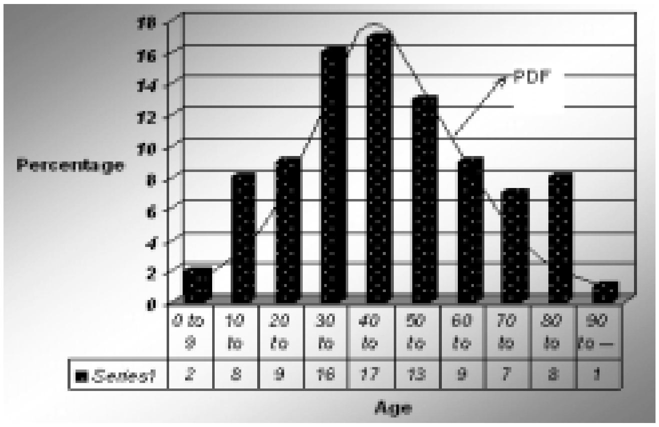


Figure 2: Producing a PDF from Sample Data

If we use “*” to denote fatality by winter storm in the bars in the Figure 2, we see that when the PDF curve reaches its peak, which means the risk probability is highest, the bar contains the highest density of “*”. In fact, the age ranges can be seen to be pre-defined clusters. So the risk of some point depends upon its cluster and unexpected points in its neighbourhood.

Therefore, we define a general risk probability as follows:

$$R(X) = \frac{U(N_r(X))}{\|N_r(X)\|}$$

where $N_r(X)$ is an r-neighbourhood of point X, $N_r(x) = \{y : \rho(x, y) < r\}$. The r-neighbourhood is not necessary a super sphere.

$\|N_r(x)\|$ is the number of all the points in this neighbourhood.

$U(N_r(X))$ is the number of unexpected points in this neighbourhood.

D. RISK PROBABILITY ESTIMATION

According to our risk probability definition, when a cluster is confirmed, the number of unexpected points in this cluster is easy to collect for historical data, from which the risk value is easy to calculate. So clustering historical interactions according to patterns is necessary as a kind of pre-processing in which distinct subclasses of patterns are discovered whose members are more similar to each other than they are to other patterns.

For each cluster, the rate between the number of unexpected results of some interactions and the numbers of elements in the cluster is defined as the Average Loss Rate (ALR). We aim to make up risk probability for each vector in this cluster by ALR. Intuitive idea is to see how close each vector X is to the average vector $\overline{X}$ in the cluster. The most common measure of similarity for two vectors is:

$$Sim(X_i, \overline{X}) = \frac{\sum_{n=1}^{m} x_{in} \times \overline{x_n}}{\sqrt{(\sum_{n=1}^{m} x_{in}^2)(\sum_{n=1}^{m} \overline{x_n}^2)}}$$

Essentially, this is Euclidean distance between two vectors. However, every element in a vector expresses a feature of the interaction. We do not know how important each feature playing a role in the interaction may be. At the same time, as all the elements have been discrete, they scatter in different ranges. The above formula cannot account for these issues. In our experiment, we give another measure of the similarity based on Mahalanobis distance [8].

$$MD_t^2 = (X - m_t)C_t^{-1}(X - m_t)'$$

Where X is one of the feature vectors in cluster t; m_t is the mean vector in cluster t; and C_t is the covariance matrix for X. It can be shown that the surfaces on which r is constant are ellipsoids that are centered about the mean m_t. In the special case where the features are uncorrelated and the variances in all directions are the same, these surfaces are spheres, and the Mahalanobis distance becomes equivalent to the Euclidean distance. It is superior to Euclidean distance because it takes distribution of the point into account. It automatically accounts for the scaling of the coordinate axes. It corrects for correlation between the different features. These properties are what we aim to achieve in our risk estimation.

Since we can make up risk probability for each vector of historical data, the trained architecture is also able to apply to new input. We briefly summarize the training procedure as follows:

- Abstract the features from the historical data to generate the features vectors;
- Using the feature vectors as input vectors, start the clustering procedure;
- After clustering, select every cluster set from the clustering results, for example: $I_k = \{V_1, V_2, ...V_k\}$,

  Here $V_i = (v_{1i}, v_{2i}, ..., v_{mi})^T, i = 1,2..., k$; I_k is a cluster, V_i is the feature vector and K is the number of the features. From next step to the end, calculate risk probabilities for every vector in this cluster. Do the same for other clusters.
- Calculate ALR:

$$ALR = \frac{\sum_{i \in I_k} E(V_i)}{|I_k|}$$

Here $|I_k|$ means the number of elements in I_k

$E(V_i) = 1$, if the result of interaction associated with V_i is unexpected. Otherwise, $E(V_i) = 0$.

- Calculate the similarity rate between any vector and the average vector-$Sim(V_i, \overline{V})$ using Mahalanobis distance:
- Calculate risk probability (RP) for every feature vector

$$RP_i = Sim(V_i, \overline{V}) \times ALR$$

The prediction procedure then comprises the following steps:

- Abstract the features from the current data to create a new vector V_l;
- New feature vector as the input vector. Start clustering procedure;
- After clustering, V_l should be in one of clusters, for example $V_l \in I_k$.
- Calculate similarity rate of between V_l and the average using Mahalanobis distance.
- Calculate risk probability (RP) for this feature vector

$RP_l = Sim(V_l, \overline{V}) \times ALR_k$, Here ALR_k is from the cluster I_k.

In order to make the prediction more precisely, each (V_i, RP_i) $i = 1,2,...k$ could be a pair (input vector, response) with proper network behavior for BP neural network.(RP_i is the response of vector V_i). Then BP neural network is trained by all the history data which is for predict new vector V_l.

E. INTRUSION DETECTION EXPERIMENT

In the 1998 DARPA intrusion detection evaluation program, an environment was set up to acquire raw TCP/IP dump data for a network by simulating a typical U.S. Air Force LAN. The LAN operated like a real environment, but being blasted with multiple attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted.

We perform an experiment to verify the above algorithms using a subset of the data which contains 34065 records. The 34065 pieces of data are divided into 24 clusters using a Kohonen Self-Organizing Map. The probability distribution for the data is given in Figure 3. Each record is just a raw TCP/IP dump and therefore every feature vector has 41 elements, denoted F1 through F41. Only 8 clusters contain unexpected connections (attacks), and the other 16 clusters are all normal connections sets with ALRs of 0. The cluster 14 is the largest one which has almost 25% connections while it contains no attacking connection. The risk of connections in this cluster is 0 that means a big part of connections are normal.

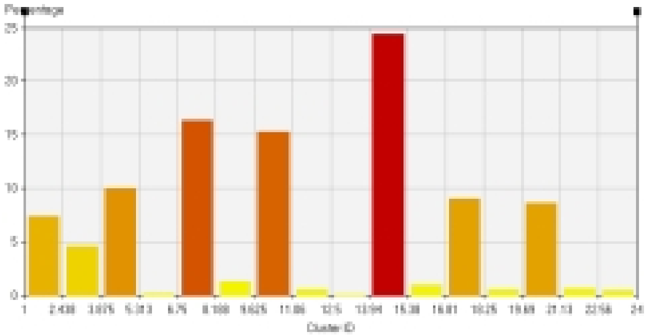
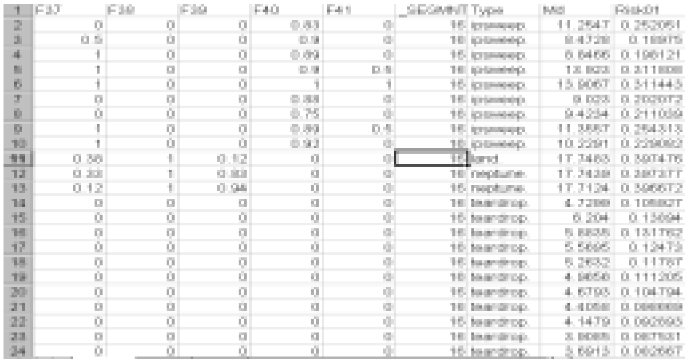


Figure 3: Distribution of clustering data

Figure 4 presents the partial results in cluster 16. The columns from F37 to F41 are elements of the feature vector extracted from the total 41 features in the raw TCP/IP data. The "_SEGMNT" column shows the cluster identification. The "Type" column describes the types of connections being made. "MD" and "risk01" describe the record's Mahalanobis distance and its risk probability. The results show that the same types of connections have similar risks, while inhomogeneous connections have distinct values.[1]



1	F37	F38	F39	F40	F41	_SEGMNT	Type	MD	Risk01
2	0	0	0	0.83	0	16	ipsweep.	11.2547	0.252051
3	0.5	0	0	0.9	0	16	ipsweep.	8.4728	0.18975
4	1	0	0	0.89	0	16	ipsweep.	8.8466	0.198121
5	1	0	0	0.9	0.5	16	ipsweep.	13.923	0.311808
6	1	0	0	1	1	16	ipsweep.	13.9067	0.311443
7	0	0	0	0.88	0	16	ipsweep.	9.023	0.202072
8	0	0	0	0.75	0	16	ipsweep.	9.4234	0.211039
9	1	0	0	0.89	0.5	16	ipsweep.	11.3557	0.254313
10	1	0	0	0.92	0	16	ipsweep.	10.2291	0.229082
11	0.36	1	0.12	0	0	16	land.	17.7463	0.397476
12	0.33	1	0.88	0	0	16	neptune.	17.7438	0.397377
13	0.12	1	0.94	0	0	16	neptune.	17.7124	0.396672
14	0	0	0	0	0	16	teardrop.	4.7289	0.105927
15	0	0	0	0	0	16	teardrop.	6.204	0.13894
16	0	0	0	0	0	16	teardrop.	5.8835	0.131762
17	0	0	0	0	0	16	teardrop.	5.5695	0.12473
18	0	0	0	0	0	16	teardrop.	5.2632	0.11787
19	0	0	0	0	0	16	teardrop.	4.9656	0.111205
20	0	0	0	0	0	16	teardrop.	4.6793	0.104794
21	0	0	0	0	0	16	teardrop.	4.4056	0.098669
22	0	0	0	0	0	16	teardrop.	4.1479	0.092895
23	0	0	0	0	0	16	teardrop.	3.9085	0.087531
24	0	0	0	0	0	16	teardrop.	3.6913	0.082667

Figure 4: Calculation of results of cluster 16

[1] However it is worthwhile to note that this is not always true, as normal connections could have the same risk as the unexpected connections. This is similar to the winter storm fatality probability example in which any principal aged 45 has the same risk of fatality in winter storm regardless of whether or not he is currently alive or dead.

We have not used higher-level features that help in distinguishing normal connections from attacks. Derived features should be able to help us to describe the interaction more clearly and briefly in order that the vector dimensions could be decreased which would lead more precise prediction.

Let us examine how this risk estimation component might be integrated with the trust-based security architecture presented in Figure 1. We cannot give risk estimation for some initial interactions for which we have no historical data. However, the monitoring component works to observe the result of the interactions and to collect relevant data to the distributed evidence store. When the data stored is enough to cover as many observed outcomes as possible, the risk evaluator is invoked to perform risk estimation using our algorithm. The historical data clustering procedure could work offline, while the estimation of risk for current interactions would work in real time as security decisions need to be made. This procedure is also similar to the human trust establishing and risk assessment. After people accumulate enough experience and identify the context and participants, they can recollect as similar as possible interactions which involves searching for data in similar context and participants to produce a precise estimation on risk.

F. CONCLUSION AND FUTURE WORK

Risk assessment a very important component in trust-based security strategy especially when the trustworthiness of some entity is unknown and no recommendation information is available. Currently, most traditional risk assessment procedures follow a fairly static process and cannot satisfy the requirements of the ubiquitous computing environment. A more flexible, dynamic risk assessment is needed.

In this paper we described a risk assessment model and proposed an estimator of risk probability that forms the core part of risk assessment in the ubiquitous computing environment. This estimator is based on a general definition inspired by traditional PDF approximation and implemented by a clustering procedure. To take the distribution of points into account, we adopt Mahalanobis distance to calculate similarity of interactions. We are currently developing the SECURE framework into which this risk probability estimator is embedded. This risk estimator is feasible and we have demonstrated that it fits well within the framework.

This project is still in progress. We do not yet know how clustering may affect the risk estimation and we have yet to establish a common evaluation process as well as common for risk estimation. However, we believe we have developed a novel way of risk estimation that can respond to new interactions dynamically and avoid subjective assessment in the ubiquitous computing environment.

Acknowledgements

This work is funded by the SECURE: Secure Environments for Collaboration among Ubiquitous Roaming Entities (IST-2001-32486), a part of the EU FET Global Computing initiative.

References

[1] M. Blaze, J. Feigenbaum, and J. Lacy. "Decentralized Trust Management. Proceedings of the 17th IEEE Symposium on Security and Privacy, Oakland, 164–173, 1996.

[2] Y. H. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick, and M. Strauss, "REFEREE: Trust management for Web applications." Computer Networks and ISDN Systems, 29(8–13): 953–964, Sept. 1997.

[3] S. Marsh. "Formalising Trust as a Computational Concept.", Ph.D. Thesis, University of Stirling, 1994.

[4] S. Fletcher, R. Jansma, J. Lim, R. Halbgewaches, M. Murphy, and G. Wyss, "Software System Risk Management and Assurance", Proceedings of the 1995 New Security Paradigms Workshop, August 22-25, San Diego, CA, 1995.

[5] V. Cahill, B. Shand, E. Gray, N. Dimmock, A. Twigg, J. Bacon, C. English, W. Wagealla, S. Terzis, P. Nixon, C. Bryce, G. Di Marzo Serugendo, J.-M. Seigneur, M. Carbone, K. Krukow, C. Jensen, Y. Chen, and M. Nielsen: "Using Trust for Secure Collaboration in Uncertain Environments", IEEE Pervasive Computing Magazine, special issue Dealing with Uncertainty, Volume 2, Number 3, pp. 52-61, Jul-Sep 2003.

[6] J. Bacon, N. Dimmock, D. Ingram, K. Moody, B. Shand, and Andrew Twigg. "Definition of Risk Model", SECURE Deliverable 3.1, 2003.

[7] http://hpccsun.unl.edu/nebraska/fatalities.html

[8] Mardia, K.V., Kent, J.T. and Bibby, J.M., Multivariate Analysis, Academic Press,1979, p31