Network Security Risk Assessment Based on Attack Graph

Lixia Xie
Civil Aviation University of China, College of Computer Science and Technology, Tianjin 300300, China
Email: lxxie@126.com

Xiao Zhang
Civil Aviation University of China, College of Computer Science and Technology, Tianjin 300300, China
Email: zx_007@126.com

Jiyong Zhang
Swiss Federal Institute of Technology in Lausanne, School of Computer and Communication Science, CH-1015 Lausanne, Switzerland
Email:jyzhang@epfl.ch

***Abstract*—In order to protect the network and evaluate the network security risks automatically, a new multi-agents risk assessment model based on attack graph (MRAMBAG) is presented. First, a network risk assessment model with master-slave agents is established, especially the functional architecture of master-slave agents and the risk association relation analysis process are designed. Then, the attack path and the attack graph are constructed by using the Attract Graph Building algorithm with the input of the dynamic data information collected by components. Finally, risk indexes of attack path, components, hosts, vulnerabilities and association risk index of network nodes are calculated successively and consequently the security risk quantitative index of target networks are obtained. The experimental results demonstrate that the MRAMBAG is a more feasible and effective way for evaluate the network security risk.**

***Index Terms*— network security, risk assessment, multi-agents, attack graph**

I. INTRODUCTION

Network security risk assessment has increasingly became one of the most essential tools for measuring the overall security of computer systems or networks, and nowadays its utilization is so widespread that it has been extensively studied. However, it has not gained an acceptable metric for network security. Involving the manual analysis of experts, the vast majority of current approaches are resource-consuming. In addition, the assessment is established only by the initial data, so it is not easy to keep a dynamic and up-to-date security metric for target networks, the most common problem of combinatorial explosion still exists.

In 1998, Phillips and Swiler proposed a method that used attack graph to evaluate network security risk [1]. Firstly, intrusive patterns and vulnerability repository were constructed according to initial specifications gathered on the target system or network environment, then the attack graph structure was modeled and visualized based on these databases. Generally, there are two dominant approaches of building attack graph by far in current research literature and commercial assessment systems: graph-theory-based attack graph assessment and model-based attack graph assessment. Model checking was firstly used to analyze whether a given goal state is reachable from the initial state [2-6] and later used to enumerate all possible sequences of attacks between the two states. In 2005, a more compact representation of attack graph was proposed based on the graph theory [7]. Based on the analysis of the security importance and security evaluation, [8] established a security evaluation system and gave the evaluation mechanism based on SVM algorithm and model. It has better time-complexity and space-complexity, but the exponential explosion in the number of such explicit attack sequences is still unresolved [9].

It is considered that these two assessments based on attack graph are insufficient to quantity network security risk and a new effective solution to build attack graph should be studied. Aimed at this, we propose the MRAMBAG that is a multi-agents risk assessment model based on attack graph which is on the basis of Amman's research. The runtime information on the target system was monitored, collected, and updated by master-slave agents. With existing descriptions of known vulnerabilities, the attack graph of component-lever was constructed which describes one potential attack, but

Manuscript received December 15, 2012; revised January 1, 2013; accepted Jun 1, 2013.

Project number: National Science Foundation of China under Grant No. 60776807 and No. 61179045, the Key Project of Tianjin Science and Technology Support Program under Grant No. 09JCZDJC16800, the Science and Technology Project of CAAC under Grant No. MHRD201009 and No. MHRD201205.

Corresponding author: Lixia Xie (lxxie@126.com).

doi:10.4304/jcp.8.9.2339-2347

many potential ways for an attacker could achieve the goal. Finally, the risk of attack path, components, hosts, the vulnerabilities and association relations risk of nodes were determined to calculate the target network quantitatively.

Ⅱ. The Framework of Risk Assessment

A. Risk Assessment Model

In this paper, we adopt attack graph to assess the overall security risk of networks. With the help of attack graphs, most of potential attack path from an attacker can be calculated [10]. Generally, the attack graph model is composed of three consecutive phases: information collecting, attack graph building and analyzing & visualizing. In the first phase, all useful information to construct attack graphs is monitored, then gathered and united including information on network structure, connected hosts, and running services. In the second phase, a graph is built based on the united databases and present vulnerability descriptions. Finally, the output is visualized to a graph structure for the further measurement.

For attack graph construction, up-to-date vulnerability information is crucial to high quality results. We address this problem by presenting the risk assessment model based on multi-agents technology. The model can dynamically update necessary information and correctly represent the current state of the network. Fig. 1 depicts a modular description of MRAMBAG. CM represents communication model.

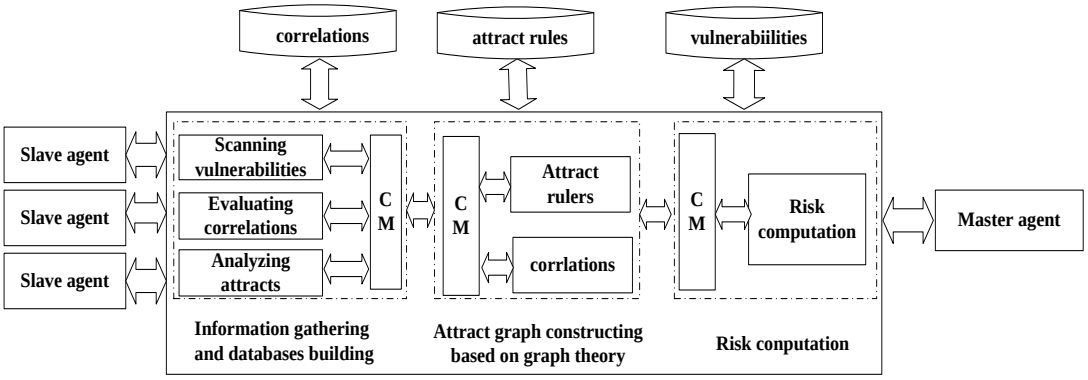

Figure 1. The structure of MRAMBAG.

In the MRAMBAG model, master agent and slave agents collect the information of network, hosts and components automatically, then databases are built based on these information. Then attack graph building algorithm based on graph theory is used to construct attack graph. And the inputs of all the potential attack paths, the risk of components, host, and network are identified by master-slave agent system. The detailed processes of evaluation are as follows.

Step 1. Start all proxy devices, and a network is scanned into catalog hosts, vulnerable network service and network connectivity. Then all the agents begin to construct the basic data cooperatively, such as vulnerability database, the attack rule database and association relations library.

Step 2. According to the database of modular attacker exploits and the specification of threat origin and known vulnerabilities, slave agents analyze all the potential attack paths on the target network and calculate the probability of a successful attack.

Step 3. Based on analysis of all known attack paths, the master agent generates an attack graph to access the overall network security, which the nodes represent components and the edges represent attack paths.

Step 4. Connect the probability of successful attack, the importance weight of hosts and components security risk, the master agent calculates and quantifies the entire host security risk. Finally the network security risk is quantified.

Step 5. Calculate and rank the risk of the vulnerability and the association relations by means of the results of the probability of the successful attack, then propose network improvements.

B. Master-slave Agent Model

Multi-agents Systems (MAS) refers to a set which is made up of a few executable agents. Every agent has the capability of network calculation. Moreover, all these agents have one same task and accomplish it with mutual coordination. To avoid the potential combinatorial explosion and collect the up-to-date information, we designed and modeled an appropriate Multi-agents System that consists of the master agent and slave agents. Its detailed parts and specific functions of the modular design are as follows.

(1) *Master Agent*.

It is mainly deployed in a dedicated or a better performance computer, and is responsible for maintaining global network information including vulnerability, attack rules, and the association, building the attack graph, and calculating the overall network security risk. Fig. 2 depicts its specific functional structure. Master Agent consists of four modules:

(*a*) *Attack rules matching module.*

Firstly Master Agent receives a request for attack rules from the slave agents, analyze them then returns the matched attack rules and appropriate vulnerability columns.

(*b*) *Association relations analysis module.*

Master Agent sums up all the association relations reports from slave agents, analyzes them, then returns the finished association relations repository to all salve agents.

(*c*) *Risk calculation module.*

On the basis of the analysis results, Master Agent calculates and ranks the risk of host components, the hosts, the whole network, known network vulnerabilities and their association relations.

(*d*) *Communication module.*

It is used on the synchronous communication among the slave agents and the master agent with structured message blocks.

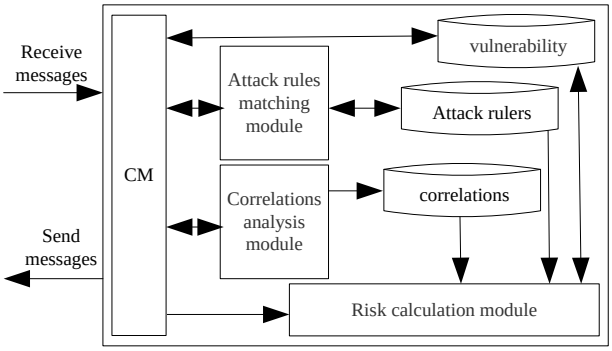

Figure 2. The function structure of the master agent.

(2) *Slave Agents.*

They are plugged in each host of the network. They take information of hosts and components as input and perform the assessment, maintain and dynamically update the local vulnerability database, the attack rule base as well as the association relation library, finally provide all useful information for evaluating network risk. Fig. 3 shows the detailed function structure. It consists of four functional modules.

(*a*) *Local vulnerability scanning module.*

Slave agents collect network vulnerabilities information including hosts, their operating systems, application programs, and vulnerable network services. Receiving the results from slave agents, the Master Agent constructs the vulnerability database.

(*b*) *Association relations Discovery module.*

Slave agents adopt the discovery tools and use the combination of automatic detection and manual analysis of the network connectivity to formulate a local association relations library.

(*c*) *Attack analysis module.*

Slave agents make a comprehensive analysis of unauthorized access that may occur in accordance with vulnerability database and association relations database, combined with the corresponding attack rule library.

(*d*) *Communication module.*

This module has the same function as master agent's communication module.

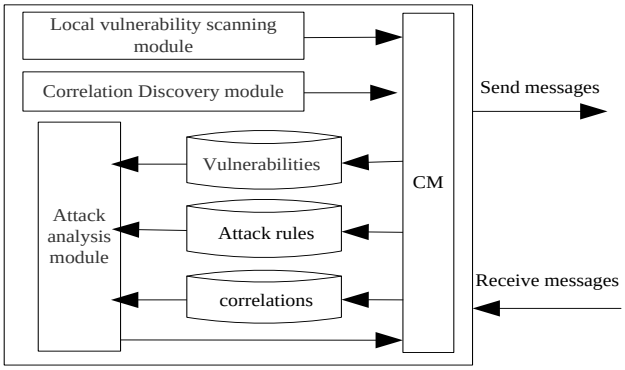

Figure 3. The function structure of the slave agent.

C. Data Collection Based on Multi-agents

In the data collection phase, the main task is collecting quantitative indicators of vulnerability, matching attack rules with vulnerability, the association relations of various components and other necessary information to construct a local vulnerability database, attack rules base and the library of the association relations. It provides the following attack analysis and the risk assessment with overall basic data. In this phase, slave agents are deployed in each host and take vulnerability list as input to generate the local vulnerability library, as well as take the trust list and visit list of various components in the hosts as input to generate the local association relations library, while preserving the attack rule base. All in all, the core of data collection and analysis is association relations analysis.

In the process of association relations analysis, slave agents construct the local association relations using a combination of automatic detection and manual analysis, and then send them to the master agent through communication module. The master agent sums up all these association relations, stores the whole network relations in the association relations library, then analyzes the data, finally returns the corresponding results with the host components to slave agents. Generally speaking, the host user knows about the components and detailed service relations, but doesn't acknowledge the global network information, so the Master Agent has to offer slave agents the necessary association relations of the network after summary and analysis. During the attack analysis process, the relations will be changeable due to the attack behavior such as illegal invasion, so the communication module should notify promptly the slave agents and the master agent to update the association relations library.

III. RISK ASSESSMENT ALGORITHM

A. Attack Graph Construction Algorithm

With the implementation of the attack graph in this paper, it becomes possible to understand what and how vulnerabilities are combined for an attack. We adopted graph theory to study and design a global attack graph generation algorithm which is used on constructing the attack graph. In this algorithm, the entire network system is represented by a graph $G = \{V, \{R\}\}$, among which V represents the set of nodes of the host components, such as the WWW service, the SNMP service and is none-empty, R represents a set of relations between two nodes which are connected by directed arc. Fig. 4 shows the main idea of the global attack graph generation algorithm.

B. Network Risk Index Calculation

In the attack graph, a node represents a host component, a direct edge between two nodes represents an attack path. Risk index calculation algorithm is performed as followings.

Input: all components' descriptions on the hosts of network, analysis results of the attack path from slave agents

Output: network attack graph

Begin

1)Listen messages from the slave agent ;

2) While (message queue is not empty) {

3) getMessage ( ); // Get a message randomly from the message queue

4) If (the message is a report about attack paths) {

5) If (there is no direct edge between the node A_i and the node B_j || the risk index is greater than the prior between the same nodes) update the attack graph edge between A_i and B_j as edge E (A_i, B_j, Attseq, P)

6) If (message queue is empty | | the time is out) send the end message of completing attack graph to slave agents;}}

End

Figure 4. Global attack graph building algorithm.

Successful Attack Probability Calculation. Generally speaking, an attack path consists of many edges and the exploit cannot be executed unless all the conditions are satisfied. If there are no proper vulnerabilities or configuration, the probability of successful attack is low, on the contrary, it is more likely to attack successfully. Therefore, the probability that whether the vulnerability in the attack path can be successfully exploited has a direct impact on the probability of successful attack. In this paper, successful attack probability is defined as P (*Attack*)

$$P(Attack) = \prod_{i=1}^{n} v_i \times p_i \tag{1}$$

where the element v_i means the i-st vulnerability that can be exploited by attackers in the attack path, the element p_i represents the probability of the corresponding successful exploited vulnerabilities.

Attack Path Risk Index (APR). Attack path is a set containing the sequences of nodes and exploited conditions reaching the attack goal. The exploited conditions not only include vulnerabilities and association relations. Also, the value of the assets and the weight in the attack path has an impact on the risk index. We formally characterize the calculation of the attack path risk index R (*AttSeq*)

$$R(AttSeq) = P(Attack) \times C \times W \tag{2}$$

where the element C indicates the value of the assets, W is the weight index, indicating the harm coefficient to the component if the attack is successful.

Component Risk Index (CR). It is important to notice that it is common multiple attack paths exist in an attack graph reaching the attack goal. Intuitively, more attack opportunities mean less security, because attackers will have a better chance to reach the attack goal. In this paper, the risk index of components R (*Component*) is the greatest risk among all attack paths. We define the calculation in (3).

$$R(Component) = \max(R(AttSeq1), R(AttSeq2), \cdots, R(AttSeqN)) \tag{3}$$

Host Risk Index (HR). Any components on a host may damage the overall security of the host. Suppose that host A consists of n components, then we can determine the weights of the host nodes according to the service distribution of the network. The host security risk index R (*Host*) is formally defined as follows,

$$R(Host) = H \cdot \sum_{i=1}^{n} R(Component_i) \tag{4}$$

where the element H means the importance index of the host in the target network.

Network Risk Index (*NR*). Because the network is composed of hosts, the risk index of the overall network can be calculated simply by summing up the individual host risk index in the network. Network security risk index is defined as R (*Network*).

$$R(Network) = \sum_{i=1}^{n} R(Host_i) \tag{5}$$

Through the risk assessment process, the local vulnerability database, the attack rule base and association relations library are real-time updated, so the key to generate the attack path is slave agents need to monitor messages from other agents, update the information and real-timely analyze the attack path.

C. Vulnerability and Association Relations Risk Index Calculation

One of the most important objectives of network security risk assessment is to provide reasonable improvement proposal to enhance network security. One advantage of MRAMBAG is adding the calculation of the risk index of vulnerability and its association relations. Attack graphs give the security personal a faster and visual understanding of the problematic pieces of a network. The attacker could obtain illegal access authority through the vulnerability or association relations. So if the vulnerability is more used, the corresponding risk probability is bigger. Therefore, we formally define the accumulation of risk indexes of all attack paths that contain the same vulnerability as the risk index of the vulnerability in (6). In the same way, the risk index of the association relations can be assessed as

$$R(Vul) = \sum_{i=1}^{n} R(AttSeq_i) \tag{6}$$

where $AttSeq_i$ indicates an attack path that contains the vulnerabilities in the attack graph. It should be noted that the value is only used to rank the vulnerabilities or association relations. Sometimes the results might be bigger than the network risk.

IV. DEVELOPMENT OF AGENTS

We implemented the Agent in MRAMBAG on basis of JADE platform (JAVA). The UML classes of agents are shown in Fig. 5.

Package jade.core included the interface Agent, class ACLMessage and class Behaviour. The Agent running on the JADE inherited jade.core.Agent and implemented the method *setup*( ). Class ACLMessage offers the communication function between Agents. Detail operations of Agent were defined in the overwrited method *action*( ) in class Behaviour. Object Behaviour was added to Agent by *addBehaviour*( ).

ChildAgent implemented interface jade.core.Agent. It analyzed the attack path according to the vulnerability setVuls, association relations NC, and attack rules. Besides, childAgent communicated with ACLMessage through methods *send*( ) and *receive*( ). The core code of childAgent is shown as follows.

```
public class childAgent extends Agent {
   List<component> components;
   List<Vul> vuls;
   List<trust> trusts;
   List<NC> visit;
   List<AttackRule> attackRules;
   protected void setUp( ) {
         ……              // initialization
      addBehaviour (new AnalysisBehaviour( ) {
         public void action ( ) {
         AttackPathAnalysis ( );
         ……
         NativeRiskCompute ( );   //calculate the local
                                    security risk
         }
      }
   }

   protected void takeDown ( ) {
      System.out.println ("childAgent"+getID( ).
                  getName( ) + "terminating");
   }
```

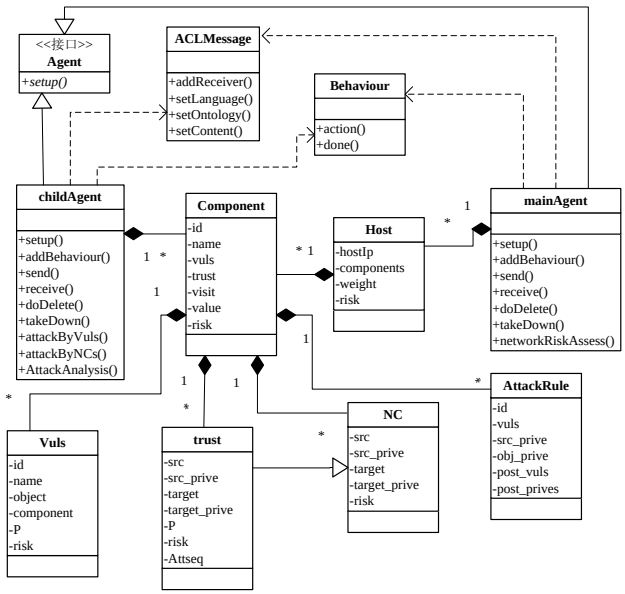

Figure 5. UMLClass of Agents.

The mainAgent implemented interface jade.core.Agent. It received the ACLMessage from the chilAgent, analyzed the message then constructed the attack graph. Also, it calculated the component and the global security risk. The core code is shown as follows.

```
public class mainAgent extends Agent {
   double NetWorkRisk;
   List<Host> hosts;
   protected void setUp( ) {
         ……              // initialization
      addBehaviour (new AssesssBehaviour( ) {
         public void action ( ) {
            AttackGraphBuild( );
            ……
            GlobaleRiskCompute( );  //compute the
                                      network risk
         }
      }
   }
}

   protected void takeDown ( ) {
      System.out.println ("mainAgent terminating");
   }
```

V. Experimental Evaluation of MRAMBAG

A. Design of Experimental Environment

We evaluated the feasibility and effectiveness of MREMAG with the method which has been used widely. A test network [11] was constructed as Fig. 6. The test network consists of host A, B, C, D, and E, respectively represents a user machine, a server, a machine in which MySQL database is running, a managerial machine and a firewall. Host A is taken as the target machine or the goal.

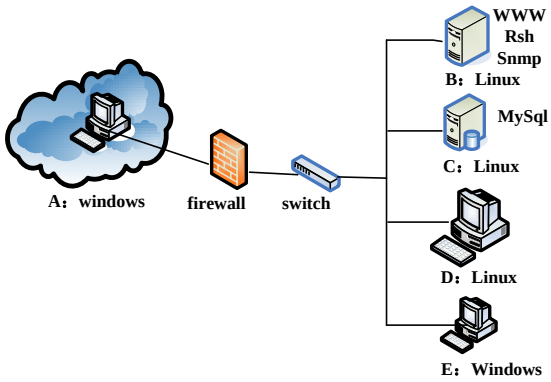

Figure 6. The test network.

The connections between nodes in Fig. 6 are as follows.

(1) The security policy of firewall is only allowing outside computer to access WWW service on Node B and all ports of node D, it will block access to other nodes and ports.

(2) Rsh service on node B set node D as a trust host, which means users on node D can execute shell commands on Node B without an authentication.

(3) WWW service on B can W/R from the database on host C but cannot manage it. Host D manages B through the Rsh and Snmp service.

(4) Rsh service of Host B run local monitor command and monitor the data flow of WWW service.

(5) Host E set D as a trust host. Users on D could login on D as a remote one.

(6) Administrator of host E can W/R from database of host C but cannot manage it.

(7) Linux, host D, has the vulnerability of buffer overflow. It allows attackers to elevate their privileges from an access one to a user.

(8) The SMB vulnerability exists in Window system of host E through which attacker could elevate their privileges to administrators. Other hosts are vulnerability less.

The connections between nodes are expressed as association relations.

(*A.any*, *root*, *B.WWW*, *access*, 0),
(*A.any*, root, *D.Linux*, *access*, 0),
(*B.WWW*, *user*, *C. MySql*, *user*, 0),
(*B.Rsh*, user, *B.WWW*, *access*, 0),
(*D.Linux*, *user*, *B. Rsh*, *root*, 0),
(*D.Linux*, *user*, *B.Snmp*, *root*, 0),
(*D.Linux*, *user*, *E.Windows*, *user*, 0),
(*E.Windows*, *root*, *C. MySql*, *user*, 0),

WWW service on B and all ports of D are open to node A that is any components of A can access these services. Therefore, the association relation of A is expressed as *any*.

Analyzing the connections between these five nodes, we obtain the initialization of list trust and visit.

$h_B.C_{WWW}.trust$ = { $h_A.C_{any}$, $h_B.C_{Rsh}$ };
$h_B.C_{WWW}.visit$ = { $h_C.C_{MySql}$ };
$h_B.C_{Rsh}.trust$ = { $h_D.C_{Linux}$ };
$h_B.C_{Rsh}.visit$ = { $h_B.C_{WWW}$ };
$h_B.C_{Snmp}.trust$ = { $h_D.C_{Linux}$ };
$hB.CSnmp.visit$ = {};
$h_C.C_{MySql}$.trust = { $h_B.C_{WWW}$, $h_E.C_{Windows}$ };
$h_C.C_{MySql}$.visit = {};
$h_D.C_{Linux}.trust$ = { $h_A.C_{any}$ };
$h_D.C_{Linux}.visit$ = { $h_B.C_{Rsh}$, $h_B.C_{Snmp}$, $h_E.C_{Windows}$ };
$h_E.C_{Windows}.trust$ = { $h_D.C_{Linux}$ };
$h_E.C_{Windows}.visit$ = { $h_C.C_{MySql}$ }.

Intruders attack against the test network using the vulnerability on D and E. The attack rule on Linux is ({Linux buffer overflow, access}, {{}, {user, visit}}). It means an attacker has the access right on all components, after attacking successfully right is elevated to user and it also acquires the access to visit list. SMB vulnerability has the same rule.

We set the importance index of B, C, D, and E respectively as 0.9, 0.7, 0.5 and 0.2 according to their importance weight in the network. The successful attack probability of vulnerabilities on Linux and windows are 0.4 and 0.7. The asset value of components is specified in Table I.

TABLE I.
Asset Values in The Test Network

Component	Values
WWW	7
Rsh	10
Snmp	5
MySql	20
Linux	5
Windows	5

The Java Agent Development Framework (JADF) we used is an open source development framework for a multi-agents system, and it has the advantage of cross-platform. We designed and developed the MRAMBAG prototype system based on JADF platform by Java.

B. Results and Discussions

The MRAMBAG performed the global attack graph generation algorithm using in the master-salve agent system constructed and visualized the attack graph by

means of the GRAPHVIZ tool. The attack graph for the given network configuration is shown in Fig. 7.

The attack graph only specified the components that relevant to illegal invasion, excluding the WWW on host B, the operate system components on host B, and host C. Node in the attack graph were connected by directed edge representing the state transfer under attack, and a curve represented illegal access relation or an attack path, and the straight line represented an illegal one.

As shown in Fig. 7, the attacker A obtain the access authority of Snmp ,Rsh of Host B, MySql component of Host C and Windows component on Host E through the vulnerability of Linux component, Host D.

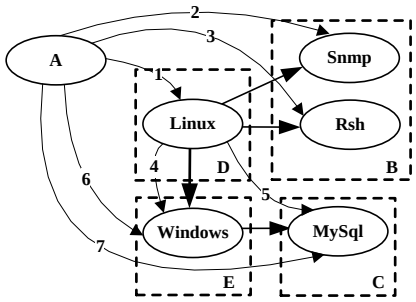

Figure 7. Attack Graph for Test network.

The attack paths in attack graph and its security risk index are shown in Table Ⅱ.

The master agent used the risk calculation algorithm to quantify the each attack path, the component nodes, hosts and the test network in Fig. 7, then the risk indexes were ranked. The risk of components, hosts, the target network and vulnerabilities and association relations are summarized in Table Ⅲ.

According to the results showing in Table Ⅲ, Host B, the network server, has the highest risk index compared to others, vulnerabilities of Linux component for buffer overflow on Host D may pose the greatest risk to the network, and the association relations that the Linux ports on Host D is completely open to Host A which is most easily exploited by intruders. Therefore in order to reduce the risk of network security, the network administrator should give priority to fix the buffer overflow vulnerability on Host D, and limit the open ports of Linux service to D on Host A.

TABLE II.
ATTACK PATHS IN THE ATTACK GRAPH

No.	Attack path	Success probability	Risk index
1	$h_A C_{any} - h_D C_{Linux}.V_{overflow}$, $NC_{A.any\ D.Linux} - h_D C_{Linux}.root$	0.4	6
2	$h_A C_{any} - h_D C_{Linux}.V_{overflow}$, $NC_{A.any\ D.Linux} - h_D C_{Linux}.root -$ $NC_{D.LinuxB.Snmp} - h_B C_{Snmp}.root$	0.4	6
3	$h_A C_{any} - h_D C_{Linux}.V_{overflow}$, $NC_{A.any\ D.Linux} - h_D C_{Linux.}\ root -$ $NC_{D.Linux\ B.Rsh} - h_B C_{Rsh}.root$	0.4	12
4	$h_D C_{Linux}.user - h_E C_{windows}.V_{SMB}$, $NC_{D.Linux\ E.windows} - h_E C_{windows}.root$	0.7	3.15
5	$h_D C_{Linux}.user - h_E C_{windows}.V_{SMB}$, $NC_{D.LinuxE.windows} - h_E C_{windows}.root -$ $NC_{E.windows\ C.Mysql} - h_C C_{Mysql}.user$	0.7	8.4
6	$h_A C_{any} - h_D C_{Linux}.V_{overflow}$, $NC_{A.anyD.Linux} - h_D C_{Linux}.root -$ $h_E C_{windows}.V_{SMB}$, $NC_{D.Linux\ E.windows} - h_E C_{windows}.root$	0.28	4.2
7	$h_A C_{any} - h_D C_{Linux}.V_{overflow}$, $NC_{A.anyD.Linux} - h_D C_{Linux}.root -$ $h_E C_{windows}.V_{SMB}, NC_{D.Linux\ E.windows} -$ $h_E C_{windows}.root - NC_{E.windows\ C.Mysql}$ $- h_C C_{Mysql}.user$	0.28	11.2

TABLE III.
EXPERIMENTAL RESULTS

NR	HR		CR		Vulnerability Risk		Association relations Risk	
	H	R	C	R	V	R	Association relations	R
27.86	B	16.2	WWW	0	Linux buffer overflow on Host D	39.4	All Linux ports on Host D are open to Host A	39.4
			Snmp	6			Host D manage Host B through Snmp service	6
			Rsh	12				
	C	7.84	Mysql	11.2	SMB vulnerability of the Windows on Host E	26.95	Through the Rsh service Host D service manage Host B.	12
	D	3	Linux	0			The user on Host D can remotely log on E.	26.95
			Linux	6				
	E	0.82	Windows	4.2			The user on Host E can access the data from MySQL on Host C.	19.6

VI. CONCLUSIONS

Quantitative risk assessment for network security has vital effects on the active protection of the network. Considering the existing risk assessment techniques are lacking of autonomy, a multi-agents risk assessment model based on attack graph (MRAMBAG) is presented to metric the overall network. The novelty of this work is adopting multi-agents technology in the risk assessment process and the global attack graph generation algorithm to automatically construct network attack graph, moreover we adopt risk assessment calculation algorithm to compute the risk indexes of components, hosts, and the target network, as well as rank the risk indexes of the vulnerabilities and association relations.

The experimental results show that MRAMBAG provides a feasible and effective way to solve the problem of quantitative assessment of the network security risk.

In the future study, based on MRAMBAG, we will devote ourselves to considering the effects of the existing safety measures and managerial factors on the network. Through repeated experiments of the real-time data, we would like to improve the assessment models in the future study and thus get more accurate metric for the network.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their detailed reviews and constructive comments, which have helped improve the quality of this paper. This work was supported in part by National Natural Science Foundation of China under Grant No. 60776807 and 61179045, the Key Project of High Technology Program under Grant No. 2006AA12A106, the Key Project of Tianjin Science and Technology Program under Grant No. 09JCZDJC16800, the Science and Technology Project of CAAC under Grant No. MHRD201009 and MHRD201205, the Central University Basic Science Research Foundation of CAUC under Grant No. ZXH2009A006.

REFERENCES

[1] Phillps C, Swiler L P, "A Graph-based System for Network Vulnerability Analysis", *Proceedings of the 1998 workshop on new security paradigms*, VA, USA: ACM Press, pp. 71-79, 1998.

[2] Ramakrishman C, Skar R, "Model-based Vulnerability Analysis of Computer Systems", *Proceedings of the 2nd International Workshop on Verification*, Pisa, Italy: Model Checking and Abstract Interpretation Press, pp. 1-81, 1998.

[3] Ritchey R, Ammann P, "Using Model C & Checking to Analyze Network Vulnerabilities", *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, Berkeley, California, USA: IEEE Computer Society Press, pp. 156-165, 2001.

[4] Sheyner O, "Scenario Graphs and Attack Graphs: PhD thesis, School of Computer Science", Carnegie Mellon University, Pittsburgh, USA, 2004.

[5] Sheyner O, Haines J, Jha S, "Automated Generation and Analysis of Attack Graphs", *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Oakland, California, USA: IEEE Computer Society Press, pp. 254-265, 2002.

[6] Jha S, Sheyner O, Wing J, "Two Formal Analyses of Attack Graphs", *Proceedings of the 15th Computer Security Foundations Workshop*, Beijing, China: Chinese Academy of Sciences Press, pp. 49-63, 2002.

[7] Ammann P, Pamuls J, Ritchev R, "A Host Based Approach to Network Attack Chaining Analysis", *Proceedings of the 21st Annual Computer Security Applications Conference*, Tucson, Arizona, USA: IEEE Computer Society Press, pp. 72-84, 2005.

[8] Yang S, Liu D, Liu Z, "Evaluating the Network and Information System Security Based on SVM Model", *Journal of Computers*, vol. 4, no. 11, pp. 1145-1150, 2009.

[9] Wu Q, Wu Y, Yang X, Hua B, Zheng R, "Evaluation of Network Connection Credibility based on Neural Network", Journal of Computers, vol. 6, no.12, pp. 2567-2573, 2011.

[10] Chen Y, Zuo W, He F, Chen K, "Optimizing Large Query by Simulated Annealing Algorithm Based On Graph-Based Approach", *Journal of Software*, vol.6, no.9, pp. 1655-1663, 2011.

[11] Zhang Y, Rang B, Chi R, "Research on Network Node Correlations in Network Risk Assessment", *Chinese Journal of Computer*, vol.30, no.2, pp. 234-240, 2007.

Lixia Xie was born in Chongqing, China in April 1974. She received the B.S. degree in Electronic Information Engineering from Harbin Engineering University, in Harbin China, in 1996. She received the M.S. degree in Software Engineering from Nankai University, Tianjin China, in 2003. Since 1996, Ms. Xie has been with the School of Computer Science at Civil Aviation University of China, where she is currently an Associate Professor. Associate Professor Xie's current major fields of study include network security and information security, network intrusion detection, cloud computation environment security and security service, intelligent information system.

Xiao Zhang was born in Henan, China in March 1989. She received the B.S degree in Shandong Construction University, in Jinan China, in 2011. Since then, she has been studying her Master degree at Civil Aviation University of China. Ms. Zhang's major fields of study include network security and information security, network intrusion detection, cloud computation environment security and security service, intelligent information system.

Jiyong Zhang was born in Hubei, China in 1978. He received his B.S. degree in 1999, and his M.S. degree in 2001, both majored in computer science, from Tsinghua University, in Beijing China. He obtained his PhD degree in Computer Science from School of Computer and Communication Sciences, Swiss Federal Institute of Technology in Lausanne (EPFL) in April 2008. Dr. Zhang has been with the School of Computer and Communication Sciences at EPFL since 2008, where he is currently a Senior Researcher. Dr. Zhang's current research interests are human computer interaction, especially intelligent user interface, recommender systems, online product search, automated decision making, e-commerce technologies, etc.