An Atomic-Domains-Based Approach for Attack Graph Generation

Fangfang Chen, Chunlu Wang, Zhihong Tian, Shuyuan Jin, Tianle Zhang

Abstract—Attack graph is an integral part of modeling the overview of network security. System administrators use attack graphs to determine how vulnerable their systems are and to determine what security measures to deploy to defend their systems. Previous methods on AGG(attack graphs generation) are aiming at the whole network, which makes the process of AGG complex and non-scalable. In this paper, we propose a new approach which is simple and scalable to AGG by decomposing the whole network into atomic domains. Each atomic domain represents a host with a specific privilege. Then the process for AGG is achieved by communications among all the atomic domains. Our approach simplifies the process of design for the whole network, and can gives the attack graphs including each attack path for each host, and when the network changes we just carry on the operations of corresponding atomic domains which makes the process of AGG scalable.

Keywords—atomic domain,vulnerability, attack graphs, generation, computer security

I. Introduction

As networks of hosts and security incidents continue to grow, it becomes increasing more important to automate the process of evaluating network security. Attack graphs are composed by all the attack paths which lead to intruders intentions. The attack path is formed by a chain of exploits, where each exploit is realized by taking advantage of known vulnerabilities in various of services and systems. The term vulnerabilities refers to exploitable errors in Configurations and server software implemented to provide network services. Essentially, attack graphs represent the security state of network, and can serve as an useful tool in several areas of network security, including intrusion detection, defense, and forensic analysis[8]. On the one hand, system administers can use attack graphs to collect informationabout their system‘s security state[9][18][23]. On the other hand, minimalcritical sets of vulnerabilities and key hosts or vulnerabilities can be computed by attack graphs[7]. Then measures can be made to strengthen the network security. This active defense can achieve better effect than thepassive defense which is achieved by collecting succeed attacks.

Supported by the National High-Tech Research and Development Plan of China under Grant Nos. 2009AA01Z438, 2009AA01Z431; the National Natural Science Foundation of China under Grant No.60703021

Fangfang Chen is with the Department of Computer Science and Technology, Beijing University of Posts and Telecommunications ,Beijing, 100876, China e-mail: lulynn_2008@163.com .
Chunlu Wang,1969,Beijing University of Posts and Telecommunications,Associate Professor,Computer Architecture and Computer Security,wangcl@bupt.edu.cn
Zhihong Tian,Ph.D,Computer Network and Computer Security,tianzhihong@software.ict.ac.cn
Shuyuan Jin,Ph.D,Computer security,jinshuyuan@software.ict.ac.cn
Tianle Zhang,Ph.D,Computer Network,tlezhang@bupt.edu.cn

Most of methods for AGG are based on privilege promotion [1][6][7][8] and information used to describe pre- and post-conditions for exploitsor attack components must be entered by hand. This is labor intensiveand difficult [20]. Now, The Common Vulnerability Scoring System (CVSS) provides an open framework for communicating the characteristics and impacts of IT vulnerabilities[17]. There are many descriptions for vulnerabilities'pre- and post-conditions. But for generating attack graph, there is an essential characteristic called relationship between pre- and post-conditions, then a vulnerabilitys post-conditions can be part of another vulnerabilityspre-conditions. Although some other vulnerability database also provideinformation relation to pre- and post-conditions, they do not contain themachine-readable details required to accurately produce many of the attackgraphs shown in past papers [20]. The lack of study on vulnerability relatednesshas been the bottleneck in attack graphs generation, analysis and usability.

The generation of attack graphs has experienced two stages including manual generation and automatic generation[7][8]. In automatic generation, model checking owns the most widespread application and approval[1][6][22]. (Such as the usage of model checking tool called NuSMV[7][8][9][22]) Besides, logic programming is also demonstrated to be an effective method owning to attack graphs automatic generation[4]. But for practical application, there are several shortcomings as follows in methods above:

1. Non-automated: For model checking, there is no transformation toolfor the networks automatic modeling[7][8][9][22]. For modern complexand large-scale network, this manual network modeling is laborintensive, difficult, and error-prone.
2. Incomplete: In model checking, the attack graphs are aiming at some host[7][8]. In fact, the administer is not sure about which host is needed to be protected, so it is essential to generate the attack graphfor the whole network.
3. Complex: All the present methods for attack graphs generation includingmodel checking and logic programming are aiming at the wholenetwork during operation. It is quit complex to deal with the informationof the whole network including all the hosts, network topology information and vulnerabilities.
4. Non-scalable: Using the present methods, the scalable generationis not achieved. When information of network are changed (such assome hosts are added or removed) then processes for generation are again.
5. Impractical: Almost all the network topology discovery tools can just achieve the getting of topology information concerning some specific host[19][20]. This makes the process of the whole network information first,generation second

unrealistic. The method of this paper provides away by which the attack graph can be generated without the whole network topology information. We just need the realization of getting topology information concerning some host.
This paper describes a new approach to attackgraphs generation which is concern about theinformation design of each host, but not the wholenetwork. This method is essentially a re-distributionof the information of network. Our method hasthe advantages as follows:
1. Simplify the process of attack graphs generation: Comparisonwith the original methods which is dealing withthe whole information of network, we are aiming at informationof each host. Besides, the design for each host is in the same template.
2. Gain the attack graphs for all the hosts of the target network: When analyzing the security of an enterprise network, it is important to consider multi-host attacks [4]. At present, there are hand generated [15], programming with some programming languages [12][13][14] and so on.. But, the methods above have poor scalability and are limited in practical applications [20]. Our approach has the ability to multiple goals attack graph generation and has good scaling.
3. Achieve the generation of attack graphs directly withoutconcerning the attack path: Previous work on attackgraphs generation all followed the order: attack paths first, then attack graphs[9][7][20]. Our method can achievethe goal without the information of attack paths. And if necessary, attack paths can be obtained by analyzing attackgraphs.

II. Related Work

Philips and Swiler[17] propose the concept of attack graph and describe a tool [21] for attack graphs generation. After that, a varietyof approaches to generating various forms of attack graphs appear, including custom-design graph-based [10][11], logical deduction[4], and model checking [1][6][7]. In model checking method, the attack paths are given as counter-examples which are related with the security policy given by network administers. NuSMV is the improvement of SMV and all the counter-examplesthat violates the safety policy can be achieved. In the prior network models, the counter-examples are attack paths corresponding in the safety policy that the intruder can not earn some privilege on some host [1][8]. By integrating all the attackpaths, the attack graph is generated [8][16][22]. Different from priorwork, our approach aims at generating the attack graph for thewhole network. In other words, we integrate all the attack paths forall the hosts in the target network. Past attack graphs are subsets of our attack graph and attack graphs for each host can be earned by reverse depth-first search from target host.
There are various of attack graphs with different kinds of nodes or edges. In paper [8][21], the nodes in their attack graphs representthe state of the network, and the edges represent an attackersactions that change the state. While in [10], their nodes of attack graphs represent an host and the edges represent the vulnerabilityused to attack next host. Taking into account the complexityof attack graphs, the former attack graphs have too many nodes, while the latter one has too many edges on each nodes. In paper[1], the authors give one design which is the compromising positionof the two illustrations above. In the initial stage, there is no limitingfactors and the attack graphs is complete that means all the attackpaths are included. Then the significant exponential explosion problemhappens. Ammann, et al. pointed out that for most computer attacks, one can assume the monotonicity property, where an attacker does not decrease his ability by launching attacks, and hence does nneed to relinquish privileges he already gained. Under this assumption, an attackers privileges always increase during the analysis. Since there are only a polynomial number of privileges an attacker can gain, the analysis algorithm will terminate in polynomial time[11]. Ourapproach achieve this by designing of atomic domains. In paper[22],the authors use an efficient semantic evaluation program in the MulVAL reasoning engine to generate the logical attack paths. They got betterrunning time than Ammanns.After that, Rattikorn Hewett and Phongphun Kijsanayothin got even better running time than paper[22] by host-centric model checking[1]. This paper achieve even better improvementthan [1] by eliminating unnecessary internal attacks in some host.

III. Attack Model

A. Definitions

A network attack model is an attack model whereall informations concerning network including network topology,hosts, vulnerabilities and so on are organized to be a finite state machine whose state transitions are based on intruders attack actions[11][17][22]. Former approaches for attack graphs generationare aimed at the whole network [1][3][6][7][8][9]. In this paper, our approach divided the network modeling into design of each host. The generation of attack graphs can be achieved by communicationsbetween atomic domains. Here are some definitions for our system:

Definition 1. $Vul =< ID, pre, post, h, s >$is a five-tupledefinition for a vulnerability. Each vulnerability has five properties as follows:
1. $Vul.ID$, the CVE standard vulnerability name [32]
2. $Vul.pre$, intruder pre-condition
3. $Vul.post$, intruder post-condition
4. $Vul.h$, the host owning the current vulnerability
5. $Vul.s$, a boolean and when valued 1(true), it means the intruder has been succeeded in capitalizing on the current vulnerability.

Definition 2. An $AD_n^h =< h, n, VN, VA, t >$ is an atomic domain for host h with privilege$n$. The following is a list of the components for each AD_n^h:
1. h, a host with the name h
2. n, the privilege for h h. There are three values for $n \in (0, 1, 2)$ representing (none, user, root).

3. VN(Vulnerabilities Needed), a set of vulnerabilities owning to h or hosts which are connected to h directly and making itpossible to achieve privilege n on host h in an exploit.
4. VA(Vulnerabilities Available), a set of vulnerabilities owningto h or hosts which are connected to h directly and being available to use when intruder achieves privilege n on host h.
5. t, a boolean variable. When valued 1(true), it means theintruder has achieved the privilege n on host h. Then attacks concerningthe set of vulnerabilities VA can be started.

Definition 3. An attack is achieved by communication between atomic domains and the communication is achieved by two actionsas follows:
1. $AD_n^h.\widehat{a}(vul)$: sending action performed by AD_n^h whose vulnerabilities set VA owns vul.
2. $AD_n^h.a(vul)$: receiving action performed by AD_n^h whosevulnerabilities set VN owns vul.
The two above are booleans and mean the success of exploit concerning vul when both valued 1(true).

Definition 4. Success label :we associate a boolean function for each vulnerability, abstractly representing whether or not the current vulnerabilityis used successfully. For the vulnerability Vul, we define the functionas $a^2(Vul)$.

Definition 5. In each attack path such as $(s_1, s_2, \cdots, s_n)$ wheres_i indicates an AD_n^h intruded during attacking, $(s_i, s_{i+1})(1 \leq i \leq n)$is called AA(atomic attack). AA has two elements as follows:
1. ss(single source), the attack sponsor in an AA.
If $(AA = (s_i, s_{i+1}))$
Then $AA.ss = s_i$.
2. st(single target), the attack goal in an AA.
If $(AA = (s_i, s_{i+1}))$
Then $AA.st = s_{i+1}$

Definition 6. NAD(Neighbor Atomic Domains) of host h is a set of AD_n^h, in which the AD_n^h owning to the host which is reachable to the current host h directly.

Definition 7. An attack model is a finite automaton $M = (AD_n^h s, a^2 s)$, where $AD_n^h s$ is a set of all the AD_n^h owning to the networkand $a^2 s$ is a set of success labels for all vulnerabilities owning to the network. The value changes of the set $a^2 s$ mean the state transformationsof the finite automaton.

B. Modeling

The process for network modeling is as follows:
1. To store the hosts newly discovered each time, we create a queue called Q.
2. Using network topology discovery tool, we get the set C of hosts which are reachable directly to the attacker and store the hosts of C into the queue$Q$. Then the vulnerabilities information of hosts in C can be got by the vulnerabilityscanning tool Nessus. Finally, the initialization of $AD_n^h s$(atomic domain) concerning attacker can be achieved.
3. Take a host from Q, and go with the following steps:
1) Using the network topology discovery tool, we get theset $\widehat{C}$ of hosts which are reachable directly to the current host.
2) Queue the hosts newly discovered in (1) into Q.
3) Get the vulnerabilities information of each host from $\widehat{C}$.
4) Initialize the $AD_n^h s$(atomic domain) concerning the current host.
4. Check whether the queue Q is empty. If Q is empty, the modelingfor network is achieved. Otherwise, go back to 3.

1) Parameter Settings: // In accordance with the definition in 2.1, for the target network with n hosts, there should be $3(n+1)$ $AD_n^h s$(including three $AD_n^h s$ concerning the intruder). But for most computer attacks, an attackerfollows the monotonicity property, where he dose not decrease his ability by launching attacks and will focus on privilege promotion [4]. This property can be valuable to the complexity reduction of attack graphs. During setting of $AD_n^h s$, we achieve monotonicity property by rules asfollowing:
1. In the initial state of the network, as the intruder owns the privilegeof administer on its own, it is not needed to notice the other two lower privileges. Then only one atomic domain is necessary for the host a owned by the attacker: AD_2^a.
2. In the initial state of the network, as the intruder owns at lest the privilege of none on each host of target network, there is no necessaryto care about the privileges of none for each host. The intruder can achievethe privilege of none on each host without attacks. Then only the $AD_n^h s$ concerning the privileges of user and administer on each host are necessary. Above all, $(2n+1)$ but not $3(n+1)AD_n^h s$ are necessary in our design. We name thehosts of the target network as $(1, 2, \cdots, n)$. Then the set $AD_n^h s$ owning the attack model M represents a set of $(2n+1)$ agents
$AD_n^h s = (AD_2^a, AD_1^1, AD_2^1, AD_1^2, AD_2^2, \cdots, AD_1^n, AD_2^n)$,
In which $(AD_1^i, AD_2^i)(1 \leq i \leq n)$are the$AD_n^h s$ concerning the host i.

2) Initialization of AD_n^h: // For atomic domain AD_n^h, the initialization of the properties$(AD_n^h.VN, AD_n^h.VA$
$, AD_n^h.t)$ concerningAD_n^h is as follows:
1. $AD_n^h.VN$: The attack towards host h can be achieved just by exploitsconcerning its own vulnerabilities. Then elements of $AD_n^h.VN$ are from host h and another condition should be: $Vul.post = n$.
2. $AD_n^h.VA$: The vulnerabilities available for AD_n^h can either come from host h or hosts which are reachable to h directly. Then the setting of $AD_n^h.VA$ is as follows:
1) For the vulnerabilities owning host h, the one which meets the rule: $Vul.pre = n$ and is not included in $AD_n^h.VN$ can be included in $AD_n^h.VA$. In our design, we prevent the redundant attacks by making sure there is no common elements in $AD_n^h.VN$ and $AD_n^h.VA$.
2) For the vulnerabilities owning hosts connected physically with host h, the one which meets the rule: $Vul.pre \leq n$ can be included in $AD_n^h.VA$. When gets higher privilege on a

host, the intruder can achieve all the exploitsconcerning the vulnerabilities with lower pre-conditions.
3. $AD_n^h.t$: When existing one vulnerability Vul in the set $AD_n^h.VN$ which meets the rule: $Vul.s = 1$, then $AD_n^h.t$ can be valued 1.
Among all the pre-conditions needed for attack graph generation, network topology is an important part, while in prior work thispart is handled as a single and complete part [1][6][7][8][9][10][16][22]. Then before generation attack graphs, the whole informationabout network topology is needed. As it is hard for the current network topology discovery tools to get the network topology, current methods for attack graphs generation are impracticalfor unknown network. In our approach, only the topology informationconcerning one host is needed and it is easy for topologydiscovery tools to achieve this. In our approach, we include the network topology information into the setting of $AD_n^h.VN$ and $AD_n^h.VA$.

C. Algorithm

After setting and initialization of $AD_n^h s$(atomic domains), the preparations for attack graphs generation have been achieved, then the process of generation can be achieved by communications between $AD_n^h s$.
Actually, our method gives the process of breadth-first penetration attacks to network . In our design, we achieve the lamination attack graph generationby the data structure queue. To the attack model $M = (AD_n^h s, a^2 s)$, in whichthe set $AD_n^h s$ represents a set of $2n + 1$ agents
$AD_n^h s = (AD_2^a, AD_1^1, AD_2^1, AD_1^2, AD_2^2, \cdots, AD_1^n, AD_2^n)$,
we can achieve the generation of attack graphsas the following:
1. To achieve the breadth-first traversal of the elements in the set$AD_n^h s$, we create a queue called Q to store the $AD_n^h s$ need to be activated.
2. Started from AD_2^a, the attacks concerning the vulnerabilities owning to $AD_2^a.VA$ can be started. Then the $AD_n^h s$ being attacked are queuedinto Q.
3. Take one atomic domain from Q (named AD_n^h), go with the following steps:
1) Active AD_n^h.
2) Start the attacks concerning vulnerabilities owning to $AD_n^h.VA$.
3) Check the target $AD_n^h s$ in (2), and queue the $AD_n^h s$ which have not been activated into Q.
4. Check whether the queue Q is empty. If Q is empty, the generation of attack graph is achieved. Otherwise, go back to 3.

D. Consideration of Implementation

SPIN is a generic verification system thatsupports the design and verification of asynchronous processsystem. Spin verification are focused on proving the correctnessof process interactions, and they can simulate the communicationsbetween processes. In our design, each atomic domain is correspondedto a process. Then the communications between ADs canbe simulated and achieved. The modeling language for SPIN is Promela(Protocol/Meta Language). Fig.1 shows the model for Promela.
In our design, we correspond the information of attackmodel M to elements of Promela as shown in Table 1.

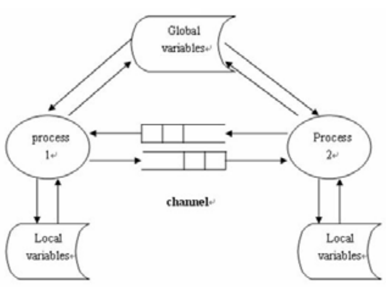

Fig. 1. Model for Promela

TABLE I
NETWORK MODEL INFORMATION

Promela elements	Information of attack model M
Global variables	Time information for each atomic domain:t; Information for communication between atomic domains; Attack matrix for attack actions between atomic domains:this part serves for monotonicity property [1][11];
processes	atomic domains;
channels	To store the communications between atomic domains;
Local variables	The elements owning to each atomic domain: $AD_n^h.n$, $AD_n^h.h$, $AD_n^h.VN$, $AD_n^h.VA$, $AD_n^h.t$;

TABLE II
INFORMATION FOR COMMUNICATIONS

sAD	The AD_n^h starting attack concerning vulnerability Vul
tAD	The atomic domain attacked concerning vulnerability Vul;
Vul	The vulnerability exploited in current attack.: this vulnerability is owned by the host concerning tAD.

Table 2 shows the information for communication stored in channels: (sAD, tAD, Vul).

Once there exits one vulnerability in $AD_n^h.VN$ which is exploited by intruder, AD_n^h is activated by valuing the property t. Then the attacks concerning vulnerabilitiesin the set $AD_n^h.VA$ can be started by AD_n^h.

IV. EXPERIMENT

Fig. 2 shows an example network used in paper[1].The network contains two hosts:web server W and database server D. An attackers host locates in the external network. The packet forwarding ortransmission is controlled by two firewalls: EF(external firewall) and IF(internalfirewall).EF allows

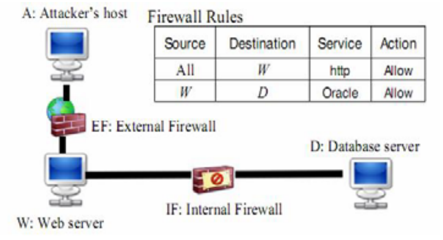

Fig. 2. Example Network

TABLE III
SUMMARY OF EXPLOITS

Vulnerability/Trust Exploit		Victim host	Pre-con. On Attack host	Post-con. On Victim host	Exploit Mode
ap	Apa.Chunked Code buff.Ovf.	W	$Access \geq 1$	$Access = 2$	Remote
tns	Oracle TNS listen buff.Ovf	D	$Access \geq 1$	$Access = 2$	Remote
t1	Trust Remote login(Any to W)	W	$Access \geq 1$	$Access = 1$	Remote
t2	Trust Remote login(W to D)	D	$Access = 1$	$Access = 1$	Remote

11.3

TABLE IV
ELEMENTS OF EACH ATOMIC DOMAIN

$AD^h_n s$	VN	VA	T
AD^a_2		$ap, t1$	1
AD^w_1	$t1$	$ap, tns, t2$	0
AD^w_2	ap	$tns, t2$	0
AD^d_1	$t2$	$ap, tns, t1$	0
AD^d_2	tns	ap	0

any packet from outside the network to be transmitted to W but IF allows only transmission from W to D but not vice versa asshown in Fig. 2.There are three hosts(Attackers host,Web Server and Databaseserver) and four Vulnerabilities(shown in Table 3) in the network.

A. Modeling for Example Network

According to the design, we can achieve the modeling for network in Fig. 2 as follows:
1. Setting of $AD^h_n s$ owning to attack model M:
$AD^h_n s = (AD^a_2, AD^w_1, AD^w_2, AD^d_1, AD^d_2)$;
2. initialization of each atomic domain :Table IV
3. Encoding : this part is achieved by the language Promela.

B. Attack Graph Generation and Analysis

By SPIN, we can get the attack graph as shown in Fig.3, in which the processes (A2, W1, W2, D1, D2) are corresponded to $AD^h_n s = (AD^a_2, AD^w_1, AD^w_2, AD^d_1, AD^d_2)$. Different from prior study, our method achieved the generation of attack

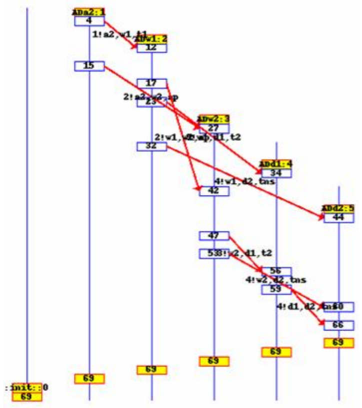

Fig. 3. Attack Graph

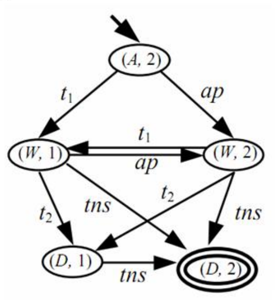

Fig. 4. Attack Graph(Advanced Host-centric)

graphwithout knowing each attack path. We achieve the generationof attack graph first, then attack paths can be get by depth-first search to attack graph. As the goal of the study is the attack graph[1][6][7][8][16][22], it is not necessary to get attack paths.
Our method generate the complete attack graph which includes all the attack path concerning each host. This is also different from previousworks which are aimed at a certain known host [1][6][7][8][16][22].

V. COMPARISON

A. Applications

Consider the network attack graph shown in Fig. 4, which is called host-centric attack graph generated in paper [1]. Host-centricmodel checking achieved lower complexity than Network-centric and access graph [1]. But in Fig. 4, there appears exploits of privilege reductionin one hosts internal attacks (such as attack from (W, 2) to (W, 1)), which does not meet the rule of monotonicity property. In our methods,we avoid this kind of situation by definition and design of atomic domains.

Fig.5 gives a new network which is achieved by adding two hosts on the network in Fig.2. The vulnerabilities for the

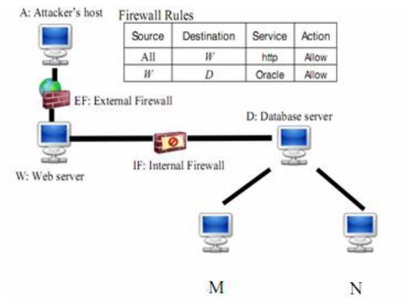

Fig. 5. Example network 2

TABLE V
SUMMARY OF EXPLOITS

Vulnerability/Trust Exploit		Victim host	Pre-con. On Attack host	Post-con. On Victim host	Exploit Mode
licq	Exploit a problem in the URL parsing function of the LICQ software for Unix-flavor system	W	$Access \geq 1$	$Access = 2$	Remote
se	The scripting action lets the intruder gain user privileges on Windows machines.	D	$Access \geq 1$	$Access = 1$	Remote
lb	Local buffer overflow	W	$Access \geq 1$	$Access = 2$	Remote
wu	WuFtpd socketprintf buff.Ovf	D	$Access \geq 1$	$Access = 2$	Remote

TABLE VI
ELEMENTS OF EACH ATOMIC DOMAIN

	VN	VA	T
AD_2^a		$ap, t1$	1
AD_1^w	$t1$	$ap, tns, t2$	0
AD_2^w	ap	$tns, t2$	0
AD_1^d	$t2$	$ap, tns, t1, licq, se, lb, wu$	0
AD_2^d	tns	$ap, licq, se, lb, wu$	0
AD_1^m		$licq, wu, tns, t2$	0
AD_2^m	$licq, wu$	$tns, t2$	0
AD_1^n	se	$se, lb, tns, t2$	0
AD_2^n	lb	$tns, t2$	0

new network are shown in Table 5.

According to the rules of ADs setting, the design for the atomic domainswhich are concerning about the two added hosts is shown in Table 6. And the attack graph for the new network is shown in Fig.6.

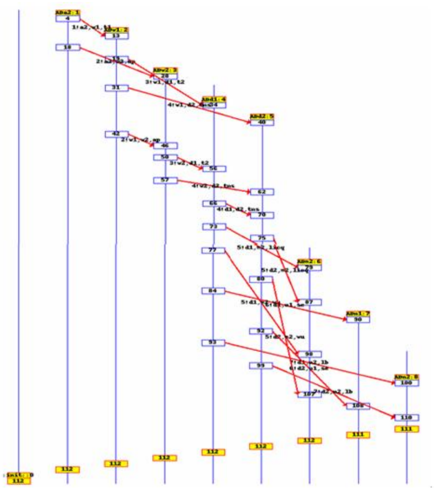

Fig. 6. Attack Graph for New Example Network

B. Discussion

Prior studies on attack graph generationfocus on the whole complex network. Once the network topologychanges, the target network must be re-modeled. Our methodprovide a way, in which we only focus on the tiny networkconcerning some host. This makes the modeling simple, scalable and reusable. This means that, we just focus on the related atomic domains whenthe topology or vulnerabilities change. Then the non-correlated part can be reused in the second modeling.

VI. SUMMARY AND FURTHER WORK

In view of present research status to attack graphgeneration, we propose a new approach which is simple and scalableby made of $AD_n^h s$. On the one hand, we just pay attention to each tiny network concerning some host. On the otherhand, when changes happen in network we just focus on the related $AD_n^h s$. The most important is that we achieve the attack graphs which includeall the attack graphs to every host owning to the target network andattack graphs for each host can be earned by reverse depth-first search fromtarget host in our attack graph.
There is some room for improvement in our design because of the following shortcomings:
1. Information redundancy: there exits information overlappingbetween $AD_n^h s$, because one vulnerability simultaneously belongs to one $AD_n^h s.VN$ property and another $AD_n^h s.VA$ property.
2. Execution randomness of processes in SPIN: the randomnessexecution of processes in SPIN makes trouble during encodingand we have to adjust the execution order of all the $AD_n^h s$.
Our future work includes information adjustment of $AD_n^h s$ andtry to different tools to achieve the $AD_n^h s'$ orderly implementation.

REFERENCES

[1] Rattikorn Hewett and Phongphun Kijsanayothin, *Host-Centric Model Checking for Network Vulnerability Analysis*, Proceedings of 2008 Annuual Computer Security Applications Conference,pp.225-234,2008

[2] Robert Richardson, *2008 CSI Computer Crime Security Servy*, http://www.gocsi.com/index.jhtml

[3] Lingyu Wang, Steven Noel and Sushil Jajodia, *Minimum-cost Network Hardening Using Attack Graphs*, Computer Communications, Volume 29, Issue 18 , 28 November 2006, Pages 3812-3824

[4] Ou X., W. Boyer, and M.McQueen, *A scalable approach to attack graph generation*, Proc. Of ACM conf. On Comp. And Com. Security,pp.336-345,2006

[5] *The NuSMV (A New Symbolic Model Checker) System*, Aval on http://nusmv.itc.it/ ,2009.

[6] Oleg Sheyner, *Scenario Graphs and Attack Graphs*, PhD thesis,Cainehic Mellon University,2004

[7] O. Sheyner, S. Jha, J. Mwing, R. P. Lippmann and J. Haines, *Automated Generation and Analysis of Attack Graphs*, Proceeding of the IEEE Symposium on Security and Privacy,pp.273-284,2002

[8] O. Sheyner and J. Wing, *Tools for Generating and Analyzing Attack Graphs*, Proceeding of Workshop on Formal Methods for Comp. And Objects,pp.344-371,2004

[9] C. Phillips and L. Swiler, *A graph-based system for network-vulnerability analysis*, Proceeding of the workshop on new security paradigms,pp.71-79,1998

[10] Paul Ammann, Joseph Pamula and Ronald Ritchey, *A host-based approach to network attack chaining analysis*, Proceeding of the 21th Annual Computer Security Applications Conference,pp.72-84,2005.

[11] Paul Ammann, Duminda Wijesekera and Saket Kaushik, *Scalable,Graph-Based Network Vulnerability Analysis*, Proceeding of ACM conference on Comp. Com.Sec.,pp.217-224,2002.

[12] F. Cuppens, *Alert Correlation in a Cooperative Intrusion Detection Framework*, Proceedings of the 2002 IEEE Symposium on Security and Privacy,Washington,DC,IEEE Computer Society,2002.

[13] P. Ning and D. Xu, *Learning attack strategies from intrusion alerts*, Proceedings of the 10th ACM Conference on Computer and Communications Security,New York:ACM Press,2003,200-209.

[14] M. Artz, NETspa, *A Network Security Planning Architecture*, M.S. Thesis, Cambridge: Massachusetts Institute of Technology, May 2002.

[15] J. P. McDermott, *Attack Net Penetration Testing*, Proceedings of the 2000 Workshop on New Security Paradings.New York:ACM Press,2001,pp.15-21.

[16] S. Jha,O. Sheyner,and J. Wing, *Two formal analysis of attack graphs*, Proccedings of the 15th IEEE Computer Security Foundations Workshop,pp.49-63,2002.

[17] *CVSS-Common Vulnerability Scoring System*, Avail. on http://nvd.nist.gov/cvss.cfm?version=2, March , 2009

[18] M. Artz, NETspa, *A Network Security Planning Architecture*, M.S. Thesis,Cambridge:Massachusetts Institute of Technology,May 2002

[19] *Network Mapper*. http://nmap.org/, 2009

[20] R. P. Lippmann and K. W. Ingols, *An Annotated Review of Past Papers on Attack Graphs*, Technical report,Massachusetts Institute of Techonology Lincoln Laboratory,March 2005.

[21] L. Swiler, C. Phillips, D. Ellis and S. Chakerian, *Computer-attack graph generation tool*, Proc. DARPA Info. Surv. Conf. Expo. ,vol.2,pp.307-321,2001.

[22] R. Ritchey and P. Amman, *Using Model Checking to Analyze Network Vulnerabilities*, Proceedings of the 2000 IEEE Symposiums on Security and Privacy,pp.156-165,2000

[23] Somesh Jha and Jeannette Wing, *Survivability analysis of networked systems*, Proceedings of the International Conference on Software Engineering,Toronto,Canada,May 2001