New Attack Scenario Prediction Methodology

Seraj Fayyad
Internet Technologies
Hasso Plattner Institute
Germany, Potsdam
Seraj.fayyad@hpi.uni-potsdam.de

Cristoph Meinel
Internet Technologies
Hasso Plattner Institute
Germany, Potsdam
Christoph.Meinel@hpi.uni-potsdam.de

***Abstract*: Intrusion detection system generates significant data about malicious activities run against network. Generated data by IDS are stored in IDS database. This data represent attacks scenarios history against network. Main goal of IDS system is to enhance network defense technologies. Other techniques are also used to enhance the defense of network such as Attack graph. Network attack graph are used for many goals such as attacker next attack step prediction. In this paper we propose a real time prediction methodology for predicting most possible attack steps and attack scenarios. Proposed methodology benefits from attacks history against network and from attack graph source data. it comes without considerable computation overload such as checking of attack plans library. It provides parallel prediction for parallel attack scenarios.**

***Index Terms*—real time prediction, attack scenarios parallel prediction, attack graph, objects oriented prediction model, learning from IDS database, new prediction methodology.**

I. Introduction

Intrusion detection systems (IDS) are used to detect the occurrence of malicious activities against IT system. Through monitoring and analyzing of IT system activities the malicious activities will be detected. In ideal case IDS generate alert(s) for each detected malicious activity and store it in IDS database.

Some of stored alerts in IDS database are related. Alerts relations are differentiated from duplication relation to same attack scenario relation. Duplication relation means that the two alerts generated as a result of same malicious activity. Where same attack scenario relation means that the two related alert are generated as a result of related malicious activities.

Attack scenario or multi-step attack is a set of related malicious activities run by same attacker to reach specific goal. Normal relation between malicious activities belong to same attack scenario is causal relation. Causal relation means that current malicious activity output is pre-condition to run the next malicious activity.

Possible multi-step attack against a network start with information gathering about network. Information gathering is done through network Reconnaissance and fingerprinting process. Through reconnaissance network configuration and running services are identified. Through fingerprint process Operating system type and version are identified.

By the end of information gathering process the attacker will have sufficient information about network to start up the attack such as:

- Network structure.
- Nodes running operating system type and version
- Node running services.

By the integration and analyzing of the collected data by information gathering the victim network attack graph could be modeled.

Possible third attack step is to identify attack plan based on the modeled attack graph in the past step. The attack plan usually will include the exploiting of a sequence of founded vulnerabilities. Mostly this sequence is distributed over a set of network nodes. This sequence of nodes vulnerabilities is related through causal relation and connectivity. Lastly Attacker start orderly exploits the attack scenario sequences till reaching his/her goal. Attack plan consist of many correlated malicious activities end up with attacking goal.

To protect the network from a multistep attack administrators usually model the attack graph of their network. Using a tool such as Mulval[14] . Attack graphs modeled by administrator can help to harden a network through finding critical vulnerabilities whose removal can prevent potential attacks [16, 17]. Attack graphs are used to monitor and predict intrusions for real-time attack responses [4, 5]. Attack graphs may also be used as a basis for designing network security metrics [18].

Attack step(s) prediction could reduce impact of the attack or prevent it. Many techniques were invented to benefit from attack graph in prediction process Such as [1, 2, 4, 7, and 12].

In this paper we propose new prediction methodology which based on the historical data from IDS and on attack graph to predict next attack step and scenario.

This paper is organized as follow: Section I is the introduction, Section II describes the related work, Section III presents our proposed prediction methodology, Section IV shows a proof of concepts of our approach and Section V concludes the finding results.

978-0-7695-4967-5/13 $26.00 © 2013 IEEE
DOI 10.1109/ITNG.2013.16

II. Related Work

Some attack prediction techniques such as [2] base on attack-scenarios library defined by experts. This library should contain all the possible attack plans against the network. When new attack scenario is run against the network the defined library will be checked for resemble defined attack plan. By the finding of a correspondence plan then next attack step(s) prediction will be implemented.

Attack-plans library should be kept up to date and filled by experts. The updating and defining of attack-plans will consume high resources. Attack plan checking causes significant overload on the prediction process.

Prediction approach such as [7] based on the extending of the alerts correlation facts. The extended facts should be in a consistence way with the encoded knowledge in the Qgraph. Prediction process in [7] predicts sets of conditions sequences from given exploit vertex in Qgraph. The resulted sequences could be considered as a set of possible condition-based attack plans. This prediction process will not identify which attack plan is the most possible one.

In [1] Interactive analysis for attack graph is proposed. [1] Proposed approach use relational model for representing attack graph input. The attack graph input data are domain knowledge and network configuration. Many data could be list under Domain knowledge such as network services, services vulnerabilities, and the relation between them. Attack graph Analyzing processes in [1] done through queries on the input data. A perfect ready to run queries is needed in this approach to predict next attack step at real time.

Some prediction techniques predict the motivation of attacker such as [14] and not next attack step. Prediction techniques in [10] use pre-knowledge about network configuration and attack to predict next attack step. [10] Prediction techniques assume that next attack step identification is based on its circumstances. These circumstances are differentiated from attacker attack preferences and network configuration. And so a pre-knowledge is needed in this techniques for prediction process

In this paper we propose new learning prediction methodology. This methodology learns from collected data by network intrusion detection system. It derive object oriented model from attack graph input data. Prediction process run based on learned data and object oriented model.

Proposed methodology does not need defined plan-library. Based on this resources consuming is reduced. Real time prediction of proposed methodology enables the real time reaction against running attack scenario. Its Parallel attack scenarios prediction attribute enable parallel protection. It predicts most possible scenario and attack step. It does not need pre-knowledge about attack scenario.

III. Prediction Methodology

We considered the Attack graph definition in [1] to be used within the proposed technology as below:

Attack graph: is a directed graph has two types of vertices, *exploit* and *condition*. An *exploit* is a triple (*hs, hd, v*), where *hs* and *hd* represent two connected hosts and *v* a vulnerability on the destination host.

A *condition* in attack graph is a pair (*h, c*), indicating the host *h* satisfies a *condition c* relevant to one or more *exploits*. There are two types of edges in an attack graph. Firstly: *require* relation which is a directed edge points from a *condition* to an *exploit*, which means the *exploit*, cannot be executed unless the *condition* is satisfied. For example, an *exploit* (h_s, h_d, v) requires following two conditions, that is the existence of the vulnerability v on h_d and the connectivity between h_s and h_d.

Second, the *imply* relation points from an *exploit* to a *condition*. It means executing the *exploit* will satisfy the *condition*. There is no direct connecting edge between two *exploits* (or two *conditions*).

Attack graph concepts are illustrated in figure 1 and formally characterized in Definition 1.

Definition 1. Given a set of exploits E, a set of conditions C, a *require* relation $R_r \subseteq C \times E$, and an *imply* relation $R_i \subseteq E \times C$, an *attack graph* G is the directed graph $G(E \cup C, R_r \cup R_i)$ ($E \cup C$ is the vertex set and $R_r \cup R_i$ the edge set).

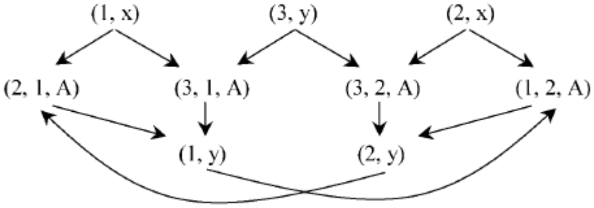


Figure 1

Definition 2: in attack graph causal relation (*CR*) is defined as forward indirect relation between two exploit nodes, which could be describe as follow:

Given a set of exploits E, a set of conditions C, a require relations (in attack graph) $Rr \subseteq C \times E$, and an imply relation (in attack graph) $Ri \subseteq E \times C$, an attack graph G is the directed graph G ($E \cup C$, $Rr \cup Ri$) ($E \cup C$ is the vertex set and $Rr \cup Ri$ the edge set).Let $x \in E$, $y \in E$ two exploits,

Ri(x): denote the set of implied conditions by the exploiting of exploit x.
Rr(x): denote the set of required conditions for the exploiting of exploit x.

CR (Causal Relation) relation is defined as follow:
$RC = \{(x, y) \in ExE : \exists c \in Ri(x) \wedge c \in Rr(y) \}$

Definition 3: We defined prediction model as object oriented based model derived from attack graph. This object model consists of a set of objects of the same class. Class of these objects is correspondence to the *exploit* vertex in attack graph. Objects of this model are connected or related using the above defined *CR* relation

Given a set of exploits vertex in attack graph E, has causal relations set CR between them.

Then Prediction model P is the directed graph P (E, CR) (E is the vertex set and CR is the edge set).

The Proposed methodology in this paper consists of set of interacted processes. These process works on a set of data come from different resources (IDS database, NVD (national vulnerabilities database) and Attack graph data source).

Tool such as Mulval[14] configure attack graph representation data in source file. This file usually contains structured data represents attack graph vertexes and relations. This data is used for the building and structuring of the proposed object oriented prediction model.

The new methodology consists of a set of correlated processes. Last output from these processes is the predicted next attack step and scenario.

These correlated processes are: model initialization process, Alerts Clustering and Aggregation, IDS alerts correlation process, mapping process (between correlated alert and proposed model objects), model causal relation weighting process and real time prediction process.

In model initialization the object model will be generated and initialized. In Alerts Clustering and Aggregation IDS database will be filtered. The filtering supposed to output one alert for one pre-happen malicious activity. It should reduce the false positive alerts as less as possible. The output of this process is the input for correlated alert process.

Alert correlated process correlate alert based on attack scenario and store its output in the correlated alerts database.

Mapping process will map each one of correlated alerts to its correspondence object in prediction model. The output of correlated alert mapping process will be used by *CR* relation weighting process. In this process forward exploit object vulnerability (in object model) revised date will be fetched from NVD [9] data base.

Lastly the real time prediction process will be run on prediction model to predict next attack step by new alert coming. *CR* relations are weighted in the *CR* relation weighting process. (Detail description about each process is given below)

Figure 2 shows proposed prediction methodology. It shows all used processes and data source. It shows also the relation between these components.

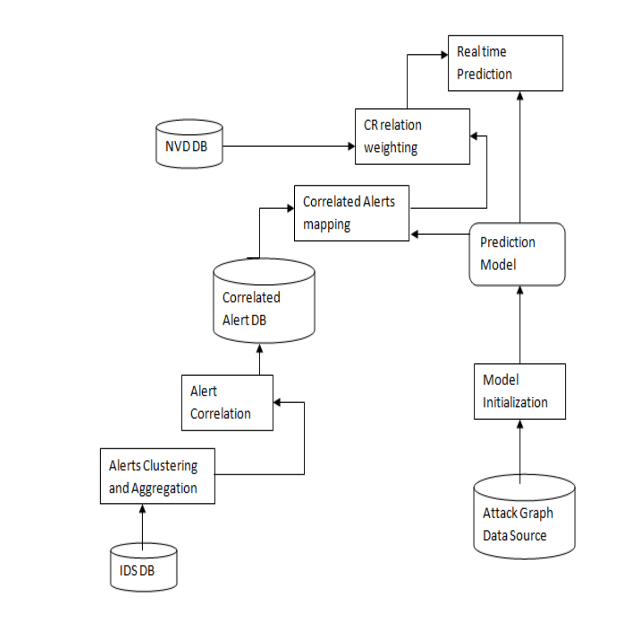


Figure 2
Intrusion Prediction Process Methodology

A. Model Initialization

In model initialization each *exploit* vertex in attack graph data source will be represented by a distinct *exploit* object. Each *CR* relation in attack graph will be represented as object to object relation in the prediction model.

B. Alerts Clustering and Aggregation:

CR relation weighting process based on *CR* relation historical number of using. We considered that the relation is used or exploit if a correspondence alert exist in IDS database.

Some factors make pure contents of IDS database contains false positive alert. Factors such as using of different IDS sensor could duplicate the alert for the same malicious activities. This duplication and false positive alerts could lead to incorrect evaluation of *CR* relation number of using.

Many techniques could be used to reduce false positive problem such as (e.g., [3] and [15]).In which alerts are grouped in different ways according to the similarity among alerts.

Some of these techniques could be considered for alerts clustering and aggregation process.

C. Alerts Correlation

In multi-step attacks attacker run first malicious activity to prepare for second one. This causal relation between malicious activities causes relations existence between related generated alerts for these activities.

Alerts Correlation process aim to find out attack scenario correspondence alerts sequence. For alert correlation many techniques are used (e.g. probabilistic based correlation, temporal based correlation and casual based correlation) as it described in [2]. These techniques work on the relation between alerts to figure out the attack scenario alerts sequence. Malicious activities of attack scenario could be identified if the alerts generated as a result of this attack scenario were identified. The output of this process should be sets of correlated alerts which reflect already run malicious activities related to a specific attack scenario.

D. Correlated Alerts Mapping

Correlation process will output sequences of correlated alerts, each one of resulted sequences is related to already run attack scenario against network. Each of these sets should be mapped separately to correspondence path in prediction model.

In mapping process each alert related exploit object is identified. And also related causal relation is identified. All correlated alerts are mapped to their correspondence exploit objects. By the end of correlated alerts mapping bordered exploit objects of *CR* relations in the prediction model will be checked. In case that the two sides exploit object has mapped alerts then *CR* relation weight automatically increased by one.

for alert mapping to correspondence exploit object in the prediction model mapping function concept in [3] is used as shown below:

t: time stamp of the alert, *s*:alert source address, *d*: destination address, *c*: class.
a single alert $a \in A$ is a tuple $a = (t, s, d, c)$ while the following functions are defined:
$ts(a) = t$ - returns $t \in T$, alert timestamp
$src(a) = s$ - returns $s \in H$, alert source host
$dst(a) = d$ - returns $d \in H$, alert destination host
$class\ (a) = c$ - returns $c \in C$, alert classification

Let *E* be the given set of exploits in prediction model and $a \in A$ set of correlated alerts belong to already run attack scenario against network:
$mapi: a \quad \{e \in E \mid \Phi i(a, e)\}$
$\Phi i(a, e) := \exists e \in E : (src(a) = src(e)) \wedge (dst(a) = dst(e)) \wedge (class(a) = ref(e))$

We characterized *CR* relation increment function as it shown below:

$\exists\ (a1 \rightarrow e1 \wedge a2 \rightarrow e2 \wedge e1CRe2)$ *imply*
RC1 weight=RC1 weight +1

E. Relation Weighting:

Before the evaluate *CR* relation weight three things should be considered:

- *CR* relations age: network is time changeable system. (Running services, Network structure…) are changeable. This time changeability in network cause differentiation in CR relation age. Some *CR* is one year old other 2 month and so on. One of the factors which used to weight *CR* relations is relation number of using by attacker. This factor should be unified per time unit. To unify this factor the number of using for each *CR* relation is considered to the same time unit. *In our study we select one year.*

  e.g. if we have CR1 age is 100 day with using time 50 times and CR2 age is 50 day with using time 40. Then: yearly using time of CR1 is 182.5 and CR2 is 292 and so using time of CR2 is bigger than CR1 using time.(in the case that relatively time unit is 365 day, which is number of days in year)

- *CR* relation using simplicity is changeable:
  Vulnerability exploiting simplicity is changeable. This changeability comes from many factors such as newly availability of a related exploiting script. A factor such past one could exchange vulnerability exploiting simplicity from difficult to simple. Vulnerability changeability causes Proportional Change of related *CR* using simplicity by the attacker.

  Some authorized resource such as NVD website [9] list the factors which affect the vulnerability exploiting simplicity.

  Important factor such *CR* using simplicity should be considered in *CR* weighting process. For this we considered that not all using times should be considered in the *CR* weighting. Just only the one which happens after last stable date of the relation.
  OR last stable of forward exploit object vulnerability.
  With new methodology the last stable date of the vulnerability is fetched from NVD [9]. In NVD last stable date called the last revised date of this vulnerability. This date should be considered as birth date of *CR* relation.

Note: last stable date of vulnerability it is the date after which no significant change on the vulnerability exploiting simplicity is happened.

- *CR* weighting should be done relatively to the other *CR* relations branched from the same object. Other *CR* relations which do not branch from same object do not connect to one of possible attack step.

After the consideration of above points we proposed following *CR* weighting algorithm:

Let:
A: set of correlated alerts.
E: set of exploits objects in the prediction model and $e \in E$
//*RC* age is calculated based on the alerts of the forwarded exploit node, where the relation is visible after the receiving of the first alert (after last revised date) related to the forwarded exploit node in the *RC* relation.//

Age (*e2*) : *RC* relation Age
{*e2* last alert date - *e2* last revised date }
relationUsed (e1,e2):{Total number of using of the relation by attackers}
//Convert Age(e2) to selected time unit (such as year)
relationUsedPerYear(*e1,e2*): *RC* relation yearly using { Time unit * relationUsed(*e1,e2*)/ Age (*e2*) }.

HEET: CR relation weight
HEET for *CR* relation has max relationUsedPerYear (*e1,e2*) between relations branched from same object is evaluated as 1.0 .

Other *CR* relations weight (which have the same root) will be calculated comparable to Max.

HEET(*e*)=relationUsedPerYear (*e1,e2*)/Max[relationUsedPerYear (*e1,e2*)]

HEET is considered as *RC* relation weights and it will be the value based on which comparisons between *RC* relation are implemented

F. Real Time Prediction:

IDS systems continuously monitor the network. By the detecting of malicious activity IDS system generate an alert and send it to IDS database. In proposed methodology IDS database will be continuously checked.. The new inserted alert will be verified and validate. Through alert registered attributes the alert will be mapped to the correspondence exploit object in prediction model. After this real time prediction process will be started. Firstly most possible next attack step will be predicted and lastly the most possible attack scenario.

- *Next Attack Step Prediction*

Proposed prediction methodology based on the object orient concept. It means that the prediction will be implemented through object interaction.

Detection of new alert insertion on IDS database mean that the related exploit object has been violated. Main goal of this step is to predict next possible violated exploit object in the prediction model. So after mapping of alert to correspondence exploit object this object is considered as prediction starting point. This object will check the *CR* weight of all connected forward *CR* relation. Based on the checked result the next forward exploit object of *CR* relation with highest weight is predicted as next attack step. We consider forward exploit object with highest *CR* weight as next attack step. Happening of other forwarded exploit object is possible also. These forwarded exploit object are differentiated in possibility percentage. Proposed methodology predicts all possible next attack step and ordered them based on their *CR* relation weight or *HEET* value.

- *Attack Scenario Prediction*

In the proposed methodology attack scenario Prediction based on also object oriented concept. Through message passing between exploits object attack scenario will be predicted. The prediction of attack scenario will be done as it describe below:

Firstly After the prediction of next attack step alerted exploit object will send prediction message to next predicted exploit object. Secondly predicted exploit object will predict its next attack step or exploit object. Thirdly it will add its Id (object deification number) to the prediction message. Fourthly it will send the prediction message to the next predicted exploit object and so on. Each exploit object receive prediction message will add its Id to message IDs list. Prediction message forwarding will be continued till prediction message is received by exploit object with no forward *CR* relation or if the message is looped. Lastly by looping or no forward *CR* relation the message will be sent back to the initiator exploit object . Message initiator exploit object will check the IDs list. Exploit objects IDs list in additional to IDs insertion order will be considered as the predicted attack scenario.

Loop identification: Each exploit object receives Prediction message will check message IDs list. If the exploit object find its id is included in the list then this mean that message was looped.. By loop last sender of prediction message is considered as the last exploit object in the attack scenario.

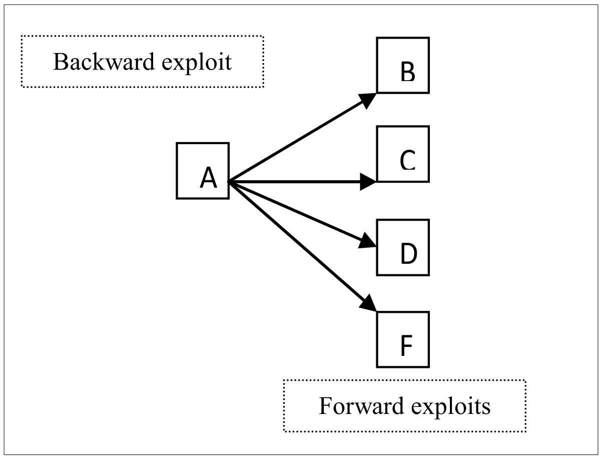
- Real time prediction algorithm:

  We characterized above concept about proposed real time prediction process with below algorithm:

```
E: set of exploits in attack graph.
Eobj: set of exploits in prediction model.
For each e in E {
Construct (e);
Eobj.addExploitObj( e)
}

Eobj.relationConfiguration();// Configure CR
relation between constructed objects. Based on
CR Definition2

//Next attack step prediction
While (true){
For each e in Eobj{
if(e.alertReceived() && e.alertVerified()){
e.PredictNextStep();
e.PredictAttackScenario();
}}

//Attack Scenario Prediction
e.Last: last exploit received prediction message at
which the message is looped or no forward step
exist
PredictAttackScenario{
sendPredictionMessage(eMax);
List IDsList= eLast.sendIDsList();
ShowPredictedAttackScenario(IDsList);
}

PredicationMessageReceiving();
{Wait (prediction message receiving){
  if (current exploit object id exist in the prediction
message IDs list)//message looped
   Send IDs list to prediction message initiator
If (current exploit has no forward exploit list) {//
exploit object with no forward CR relation
Add current exploit id to IDs List
Send attack scenario IDs to prediction message
initiator
}Else
PredictNextStep()
Add object id to the message IDs list
Send prediction message to next predict exploit
object
}
```

IV. Proof of concepts

Consider the following graph as a part of derived prediction model from attack graph where *A* is considered as backward exploit object and *B, C, D, F* forward exploits objects:



With below example we assume that: number of using for all *CR* relations after stability status was evaluated as it was described. *CR* relations ages was calculated using given description in the *CR* relation weighting algorithm. And the yearly using for each relation in additional to weight (HEET) was calculated using *CR* weighting algorithm.

As it shown in the below table *HEET* are calculated and from it the most possible next attack step is identified..

HEET EXAMPLE

CR Relation	*Using times*	*Age (day)*	*Yearly using*	*HEET*
AB	17	610	10.17	0.167
AC	50	300	60.83	1.0
AD	10	234	15.60	0.25
AF	20	435	16.78	0.275

The next attack step is the exploiting of C exploit object. Where AC using times is the biggest between other branched *CR* relations and so it has biggest HEET which is 1.0.

V. Conclusion and Results

We propose a new real time prediction methodology. Which enable the real time protection against running attack scenario.

This methodology learns from IDS database and improve prediction quality with time. It predicts all possible next attack steps and ordered them based on their possibility.

Proposed approach offer parallel prediction for parallel multi-step attack using object oriented concepts. If two multi-step attacks are run parallels against different goals within network then for each one a distinct prediction process will be run.

REFERENCES

[1] Lingyu Wang a,∗, Chao Yaob, Anoop Singhal c and Sushil Jajodia dImplementing interactive analysis of attack graphs using relational databases, in: Journal of Computer Security 16 (2008) 419–437 419, DOI 10.3233/JCS-2008-0327, IOS Press

[2] Qin, X.: A Probabilistic-Based Approach for INFOSEC Alert Correlation, PhD thesis, Georgia Institute of Technology (2005)

[3] Sebastian Roschke, Feng Cheng, Christoph Meinel: A New Correlation Algorithm based on Attack Graph In Proceedings of the 4th International Conference on Computational Intelligence in Security for Information Systems (CISIS 2011), Torremolinos, Spain, June 2011

[4] L. Wang, A. Liu and S. Jajodia, An efficient and unified approach to correlating, hypothesizing, and predicting intrusion alerts, in: Proceedings of the 10th European Symposium on Research in Computer Security (ESORICS 2005), LNCS, Vol. 3679, Springer-Verlag,Milan, Italy, 2005, pp. 247–266.

[5] L. Wang, A. Liu and S. Jajodia, Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts, Computer Communications 29(15) (2006), 2917–2933.

[6] Steven Noel, Sushil Jajodia, "Understanding Complex Network Attack Graphs through Clustered Adjacency Matrices," acsac, pp.160-169, 21st Annual Computer Security Applications Conference (ACSAC'05), 2005

[7] Wang, L., Liu, A., Jajodia, S.: Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts. Computer Communications 29(15), 2917–2933 (2006)

[8] First: WEBSITE, http://www.first.org/cvss (accessed May 2012)

[9] NVD: WEBSITE, http://web.nvd.nist.gov/view/vuln/search (accessed May 2012)

[10] Sanjeeb Nanda and Narsingh Deo. The Derivation and Useof a Scalable Model for Network Attack Identification and Path Prediction

[11] Nexat: a history-based approach to predict attacker actionsACSAC '11 Proceedings of the 27th Annual Computer Security Applications

[12] L. Wang, A. Liu and S. Jajodia, Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts, Computer Communications 29(15) (2006), 2917–2933.

[13] R. Deraison, Nessus scanner, available at http://www.nessus.org, 1999.

[14] R. Katipally, L. Yang, and A. Liu, “Attacker behavior analysis in multi-stage attack detection system,” in Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research, New York, NY, USA, 2011, pp. 63:1–63:1.

[15] JULISCH, K. and DACIER, M., “Mining intrusion detection alarms for actionableknowledge,” in The 8th ACM International Conference on Knowledge Discovery andData Mining, July 2002.

[16] S. Noel, S. Jajodia, B. O’Berry andM. Jacobs, Efficient minimum-cost network hardening via exploitdependency graphs, in: Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC’03), Las Vegas, NV, USA, 2003.

[17] L. Wang, S. Noel and S. Jajodia, Minimum-cost network hardening using attack graphs, Computer Communications 29(18) (2006), 3812–3824.

[18] L. Wang, A. Singhal and S. Jajodia, Measuring the overall security of network configurations usingattack graphs, in: Proceedings of the 21th IFIP WG 11.3 Working Conference on Data and Applications Security (DBSec 2007), Redondo Beach, CA, USA, 2007.