Context Model Fusion for Multistage Network Attack Simulation

Stephen Moskal, Ben Wheeler, Derek Kreider, Michael E. Kuhl, and Shanchieh Jay Yang
Rochester Institute of Technology, Rochester, New York 14623

***Abstract*—Analyzing and predicting complex network attack strategies require an efficient way to produce realistic and up-to-date data representing a variety of attack behaviors on diverse network configurations. This work develops a simulation system that fuses four context models: the networks, the system vulnerabilities, the attack behaviors, and the attack scenarios, so as to synthesize multistage attack sequences. The separation of different context models enables flexibility and usability in defining these models, as well as a comprehensive synthesis of attack sequences under different combinations of situations. After describing the design of the context models, an example use of the simulator and sample outputs, including the ground truth actions and sensor observables, will be discussed.**

I. Introduction

The large volume and sophistication of cyber attacks on enterprise networks have demanded advances to enhance Cyber Situation Awareness (Cyber SA). Since early 2000's, much research has worked on investigating how to comprehend complex, multistage attack strategies, *e.g.,* [1], [2], [3] and anticipate future attack actions, *e.g.,* [4], [5], [6]. While these works provided novel and promising ideas for Cyber SA, evaluations of the effectiveness have been sporadic and ad hoc; thus, little consensus can be built on how to best estimate and predict multistage attacks.

The lack of formal processes and commonly available data for evaluating Cyber SA work is not without reason. Generating real data sets that cover multistage network attacks with a spectrum of hacking skills on a variety of sufficiently large network configurations would be extremely resource consuming. In addition, the data would need to contain ground truth of the attack strategies and the observables based on the types and placement of sensors on the network. This would require highly skilled personnel to set up the network and execute and document the attacks. Perhaps the most daunting task among all is to continuously update the network and attack strategies, to sufficiently reflect new vulnerabilities, exploits, and attack strategies. This work aims at alleviate these challenges by developing a flexible simulation system, called Multistage Attack Scenario Simulator (MASS).

Cyber attack simulation is not entirely new. In late 1990's, Cohen developed one of the first cyber attack simulators [7]. His simulator used a simplified computer network model and a cause-and-effect model [8] consisting of 37 threat profiles (behaviors), 94 attacks (physical and cyber), and 140 defense mechanisms. While this pioneering work demonstrated how known attacks could penetrate and impact networks, it was limited to pre-defined models and would require significant effort to update. Following Cohen's work, Park *et al.* [9] created a simulator named SECUSIM in 2001. SECUSIM allowed user defined attack mechanisms and defense strategies, but its implementation of high-level attacker behavior required predefined attack paths. Neither of these simulators accounted for specific vulnerabilities or generated sensor outputs. Kotenko and Man'kov [10] defined a tool called Attack Simulator in 2003, which used probabilistic state machines to guide attack generation. Like SECUSIM, the attack generation relied on high-level attack steps such as Resource Enumeration, Privilege Escalation, and Back Door Creation to guide the attack generation. Additionally, very little information about the target network was definable and accounted for in the simulation algorithm. A different type of simulator was developed by Santhi *et al.* [11] to model malware propagation in online social networks by accounting for latest published vulnerabilities.

The aforementioned simulators were not designed for the purpose of generating data to evaluate Cyber SA technologies, and can not efficiently support varieties and changes to the attack behaviors, network configurations, and system vulnerabilities. Note that supporting Cyber SA means the need of both ground truth attack activities and realistic sensor outputs. To this end, we develop MASS, consisting of a attack generation core that fuses four context models: Virtual Terrain (version 2), Vulnerability Hierarchy, Scenario Guiding Template, and Attack Behavior Model. In a way, MASS builds on the strengths of the aforementioned simulators, and expands the concept of two context models, Virtual Terrain [12] and Guidance Template [1] used in the simulator developed by Kuhl *et al.* [13]. A high-level preliminary introduction of this work was presented in a poster-paper [14]. The following sections discuss the details of MASS and demonstrate its capabilities through attack scenario examples.

II. Context Model Fusion for MASS

Multistage Attack Scenario Simulator (MASS) comprises an attack generation core and four context models: Virtual Terrain v2 (VT.2), Vulnerability Hierarchy (VH), Scenario Guidance Template (SGT), and Attack Behavior Models (ABM), as shown in Fig.1. MASS simultaneously generates multiple attack sequences and the corresponding sensor observables with noise. The attack sequences are generated based on randomized selections of exploits within the conditions specified in the stage-based scenarios defined in the SGT and attack preference parameters defined in the ABM. The observations, or non-observations, of the attack actions are generated depending on the system vulnerabilities and sensor placements and capabilities defined in the VT.2. Zero-day attacks can be simulated without producing observables. The attack generation core combines/fuses the context models, which will be specified by experts who have knowledge of them (adversary) and knowledge of us (network), respectively. MASS and its context models are designed to provide flexibility and scalability

to generate a large number of attack sequences even with imperfect or incomplete context model specifications.

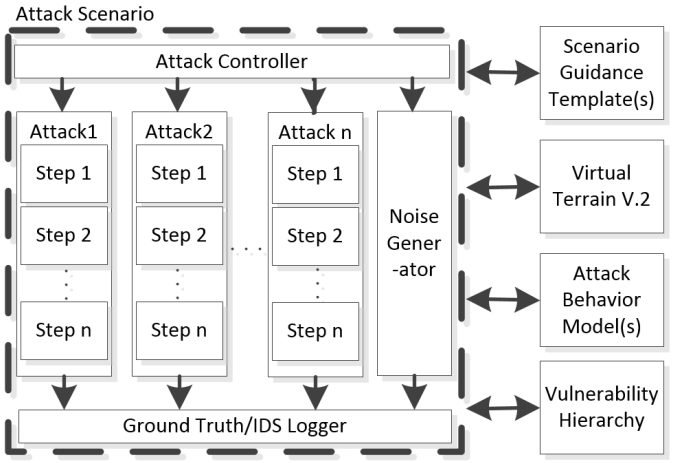

Fig. 1: An architectual view of MASS (revised from [14]).

A. Virtual Terrain v2

The computer network model used in this work is an extension of the Virtual Terrain defined previously by Holsopple *et al.* [12] and is accordingly named Virtual Terrain v2 (VT.2). The new model is designed to provide flexibility and generality, recognizing that a user may not have full knowledge of the entire network configuration, or wish to randomly generate various network configurations.

In VT.2, a host machine or a server is generically defined as a node. A node contains attributes that are relevant to cyber attack scenarios, including IP address, send and receive permissions, services in use, intrusion detection system (IDS) or intrusion prevention system (IPS) sensors (if any), and Internet visibility (if any). A permission set from node i to node j means that an IP packet is allowed to come from i to j, and, thus, malicious action is possible to happen. Note that spoofed IP will be accepted as long as the source IP is permitted by the target node. Services defined for a node is used in conjunction with VH to determine the corresponding vulnerabilities and exploits. Zero-day exploits are built into the VH to allow the simulation of such possibilities. A node can have zero, one, or multiple sensors, each of which can be defined to include the set of exploits it can detect and its range. The sensor range is defined to include the set of links entering or leaving the node the sensor is attached to.

When a set of nodes has identical configuration, a node-group can be defined to efficiently represent the attributes. The node-group reduces the computation and storage needs when using VT.2. VT.2 also specify a special "router" object. A router is a node without defining service vulnerability. It serves as a pass-through between nodes, to allow easy traversal of VT.2 when determining likely targets of next attack actions. An actual critical router in the network with vulnerabilities can be modeled as a node. Another special object in VT.2 is the "Internet," which allows the random generation of external IPs as source IP attacking the network.

Users do not need to specify all attributes for all nodes and routers in the network; the attributes can be either specified as any, none, or randomly generated. In fact, a user can simply specify a set of services in the network and the IP ranges, and a customized tool will generate an appropriate VT.2 for simulation purposes.

Figure 2 summarizes the VT.2 definition. The router, subnet, and node objects are depicted with different shapes. The attributes of each object are shown in the 4-tuple form of $(IP, \{S\}, \{N\}, V)$, where IP is the IP address or address range, $\{S\}$ is the set of services (if applicable), $\{N\}$ is the set of sensors, and V is a binary flag indicating the visibility of the object directly from the Internet. The directed edges represent the access permissions from one object to another in the network.

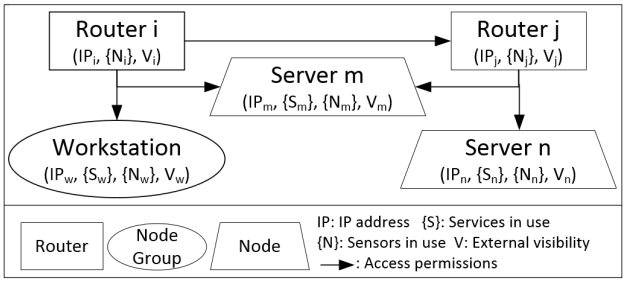

Fig. 2: Graphical summary of VT.2 scehma.

B. System Vulnerability Hierarchy

The services specified in the node objects within VT.2 need to be mapped to a known set of vulnerabilities, and, consequently, the corresponding exploits and sensor outputs. In a generic context, we chose to use a hierarchal structure to describe this mapping. Each level in the structure describes a vulnerability in more detail as the levels increase: the top level of the hierarchy contains a few high-level descriptions of attack types, whereas the bottom level of the structure contains specific and unique vulnerabilities of different systems. This structure can be a minimum of two levels, restricting to just services and vulnerabilities. There is no maximum size to the hierarchy due to its generic set up; the user only needs to define the number of levels in the structure and what level the services are located.

Note that separating the modeling of the network (VT.2) and vulnerabilities (VH) allows the users to focus on developing the network structure for simulation, without concerning the most recent known vulnerabilities. For this work, a comprehensive and up-to-date database of cyber system vulnerabilities is implemented to accurately describe and simulate multistage network attacks. This gives the capability of the simulator to have relevant results with respect to the industry reported known vulnerabilities as well as the potential zero-day attacks. In terms of industry reported known vulnerabilities, this work chose to reference to NVD (National Vulnerability Database) by NIST (National Institute of Standards and Technology) due to its daily comprehensive updates along with readily available and consistent data.

Each vulnerability in the NVD is assigned a specific unique value by the year it was discovered and its position on the list of vulnerabilities (*e.g.,* 2014-0001), which is called a CVE (Common Vulnerabilities and Exposures) managed by the MITRE Corporation. For each CVE, another value is assigned to it called a CWE (Common Weakness Enumeration) which

gives a higher level classification of the vulnerability along with a set of CPE's (Common Platform Enumeration) which describes the services that is vulnerable to that specific CVE. As of February of 2014, the NVD contains 60324 CVE's and 83708 CPE's and the MITRE database contains 940 CWE entries and 19 categories of CWE's.

Additional sets of unknown vulnerabilities are defined in a generic fashion for the CPE's to enable the simulation of zero-day attacks. Recon-type of malicious activities are also defined to complement NVD and to enable the comprehensive simulation of different network and system scanning activities that often accompanies vulnerability exploits. The standard CVE's and recon activities are mapped to specific types of sensors, *e.g.,* Snort. Note that not all CVE's are mapped to Snort alerts, which is part of what the simulator wants to exhibit.

The goal of the Vulnerability Hierarchy (VH) is to structure the data in which the top level is a high level abstraction of attack types and then the actual vulnerabilities at the bottom of the data structure. The implementation of VH in this work contains four levels: 1) the categories of CWE's, *e.g.,* "Read files or directories" and "DoS: Resource Consumption," 2) the CWE level, *e.g.,* "Improper Restriction of Operations within the Bounds of a Memory Buffer" and "External Control of System or Configuration Setting," 3) the service level or the CPE level, providing the mappings between the services defined in VT.2 and the VH, and 4) the CVE level. Figure 3 shows a graphical representation of a small subset of the VH.

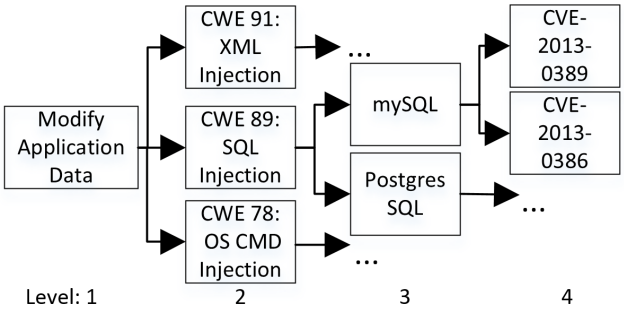

Fig. 3: A graphical representation of a small subset of VH.

C. Attack Behavior Model

Simulating multistage network attacks requires not only the context models of the network and the vulnerabilities, but also the context model of what attackers might do. Authentic and comprehensive attack behavior modeling is extremely challenging, and no existing work truly provided a viable approach. Related work on attack behavior modeling includes those that model how network attacks could transpire over time, but do not exactly capture the interactive nature of attacker behavior. Such attack modeling works include the attack graph based approach by Jajodia *et al.* [2], the Bayesian Network based modeling by Qin and Lee [4] and by Xie *et al.* [15], the graph template based modeling by Stotz and Sudit [1], and the Variable Length Markov Model prediction by Fava *et al.* [5]. The works that attempted to model the "attacker behaviors" include the modeling of adversary capability and opportunity by Holsopple *et al.* [16], the use of Markov Decision Process by Zhang *et al.* [17], and the use of Game Theory by Wang *et al.* [18].

Unfortunately, none of the above approaches is structured to generate interactive attack actions based on what the attacker learned through the outcomes of past actions, which requires the modeling of the services, vulnerabilities, and data in the network. The current implementation of the Attack Behavior Model (ABM) is also limited. As a step towards a comprehensive attack behavior modeling, this work separates the parameterized definitions of attacker capabilities and preferences (specified in ABM) from the modeling of desired attack scenarios (SGT), which will be described in the next section.

Table I shows the parameters used in ABM to describe attacker skills and preferences. Specifically, Stealth Skill describes how likely an attacker uses an exploit that is not specified in the VH or the sensors. A high Stealth Skill will have more attack actions go undetected to represent either zero-day attacks. Discovery Skill represents the likelihood of an attacker exploiting or finding a node that was determined hidden or not accessible as specified in VT.2. A low Discovery Skill will require the attacker to conduct more recon actions to discover the accessible nodes from the set of nodes the attacker currently has control. The Average Trials describes, on average, how many times an attacker will attempt one or more exploits on the same target before giving up. A large Average Trials reflects potential "script-kiddies" who may trigger many sensor alerts because of high volume of attempts.

TABLE I: Parameterized Attributes in ABM.

Attribute	Description
Stealth Skill	Probability that the attack step goes undetected.
Discovery Skill	Probability that the attacker directly attacks a hidden node.
Average Trials	The average number of times the attacker will attempt an attack step before giving up.
Attack Skill	The attacker skill level used to determine whether an attempted exploit will be successful.
Preferred Attacks	A list containing the preferred attacks from the VH entries.
Preferred Nodes	A list containing the preferred nodes on the network.
Specific Preference	A specific set of preferred attacks on a per-node basis.

The Attack Skill attribute is designed to determine whether an attempted attack action will be successful by comparing it to the "skill" value (S_v) specified for each CVE. The S_v used in CVE is a number between 0 and 10, representing how difficult to exploit a particular CVE as determined by NVD; the higher it is, the more difficult for an attacker to exploit the vulnerability successfully. The current implementation uses a probability function, shown below, to reflect that the chance of an attack gets exponentially more difficult to succeed (but not zero) as Attack Skill (S_a) decreases below (S_v) for a given CVE v.

$$\mathcal{P}(\text{success}) = \begin{cases} 1 & \text{if } S_a \geq S_v \\ \mathrm{e}^{-(S_v - S_a)} & \text{if } S_a < S_v \end{cases}$$

The last three attributes, Preferred Attacks, Preferred Nodes, and Specific Preferences, are optional and allow the user to specify any preference the simulated attacker might have with respect to attack types, services, IP addresses, or a combination of them. The user can further specify weights to the preferences if there are multiple ones. For example, the

user may specify the preference to MySQL exploits, or to a specific domain server on the network. In the extreme case where the user wants to simulate a case where the attacker knows the network extremely well, *e.g.,* an insider attack, the Specific Preferences can be used to define specifically which exploits to be used on which nodes in the network.

D. Scenario Guidance Template

ABM specifies the "preference" and "skills" of the attacker, and will need to be complemented by a context model that specifies the desired attack scenarios, including the stages and transitions of the desired multistage attacks. Scenario Guidance Template (SGT) is designed to achieve this goal. Attack scenario modeling can draw from the aforementioned attack modeling works [2], [4], [15], [1], [5]. Particularly, this work chooses to adopt the concept of graph-based template by Stotz and Sudit [1] and build on the work by Kuhl *et al.* [13] to define SGT. This approach allows separating the modeling of desired attack scenarios from the specific network topology and configurations.

The SGT allows defining one or multiple scenarios for a single simulation run. It controls the potential attack progression from the beginning to the end of the simulation and represents the progression as a graph, where each node reflects a stage of the attack and each arc represents the condition that must be met in order to transition from one stage to another. Within each stage of an attack, a user can specify if the stage is a starting point, the attacker's goal stage, or if the transition to that stage will cause the attacker to choose a new target within the network. The arc conditions can be expressed using the attack categorization specified in VH, based on the state of the target machine after performing the attack action, or simply whether the exploit performed was successful or not. Unconditioned (default) arcs can be used to handle unspecified situations. The user can also restrict a stage to limit the simulator to choose from exploits that will satisfy the outgoing arcs from that stage. Throughout the simulation run, the simulator keeps track of the current stage of the SGT and transitions when an arc condition is satisfied after an attack step is performed. If an exploit is performed within the goal stage that results in the compromise of a target machine, the attack scenario will terminate and the results will be stored in both ground truth and sensor output files.

Figure 4 shows a graphical depiction of an SGT of an SQL attack. This attack scenario is composed of three stages, starting with a restrictive information gathering stage limiting the simulator to output either a reconnaissance type of attacks specified by arcs (1,A) and (2,B) or a DoS attack specified by the arc (1,C). During this stage of the attack if the current target machine has not yet been exposed by the attacker the simulator will likely attempt a reconnaissance type of attack starting at the most specific level, if possible in this case, an exploit contained within Network Scan. If the target machine was already exposed or the attacker was specified as highly skilled in the ABM, the simulator would attempt a DoS attack. If this DoS attack was successful the simulator would transition from the first stage to the third stage directly via arc (1,C). In the case of an unsuccessful attack, the stage would remain the same and the attacker could choose a different target machine from his current location in the network or back off to a previously compromised machine if necessary. The second stage DoS:Intrude continues the attack progress to the DoS stage if the first stage only accomplishes information gathering through reconnaissance type of attacks. The final stage of the scenario is a non-restrictive SQL attack stage where the exploits generated are not limited to the ongoing arc conditions. If the exploit is a MySQL attack and successful, the simulated attack sequence ends successfully; if not, it will continue to try a maximum of times as specified in ABM or retreat to the first stage based on the default arc.

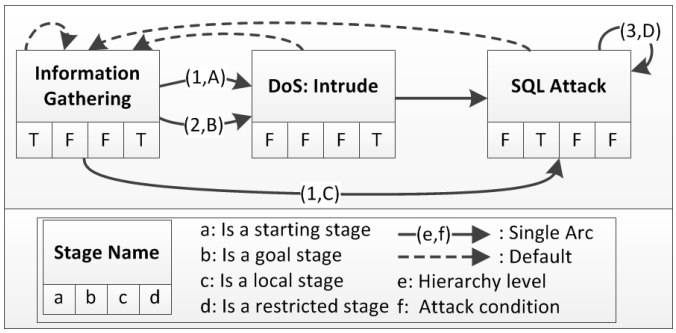

Fig. 4: An example SGT for SQL attacks. (A: Reconnaissance, B: Network Scane, C: DoS Attack, D: MySQL Services)

III. Sample Outputs of MASS

This section presents an example of how MASS simulates an attack scenario and produces the ground truth of the multistage attacks and the corresponding sensor outputs.

A. Configuration

The target network for this attack scenario contains thirty nodes, and is illustrated in Figure 5. There are two main entry points from the Internet into the network: Router 1 and Router 2. Router 1 provides access to three workstations running Mac OS X, while Router 2 provides access to five workstations running Linux. Each of these eight workstations is connected to Router 3 and Router 4. Router 3 provides access to ten workstations running Windows. Note that one of Windows node is visible directly from the Internet, making it a potential third entry point into the network. Router 4 provides access to three servers: Directory Server, Domain Server, and Web Server. Router 3 and Router 4 are also connected to Router 5, which provides access to four additional servers: Oracle Server, DB2 Server, MySQL Server 1, and MySQL Server 2. The routers in this network are assumed as pass-through machines where attackers from one side of the router can directly attack a node on the other side.

The ABM defines an attacker who prefers MySQL injections. The "Stealth Skill" and "Discovery Skill" are set very low (.001) to encourage data that abide by the network configuration defined in VT.2. If these values are set high, the data may not be reflective of the network constraints because the attack can bypass many steps and traverse through the network easily. To encourage successful attack actions, the Attack Skill is set to 8 (mean skill level is 6 for the CVE's considered). The average timeout is set to 3. Only the preferred attacks are used to promote the exploitation of MySQL service.

The desired outcome of the simulated attack sequence is a successful SQL attack on any node in the network. The

SGT specifies a three-stage attack scenario. The first stage aims at gather information on accessible targets via network scan. The second stage specifies a successful DoS attack to infiltrate the network. The SGT will allow attack to bypass the first stage and directly perform DoS successfully with a probability. Once successfully infiltrate a victim machine that has MySQL service or has access to a machine that runs MySQL, the last stage is to exploit the MySQL vulnerabilities of that MySQL server. If at any point during the execution of the attack sequence the attack action fails, the sequence will return to the first stage of SGT and attempt to progress along a different path in the network.

B. Results

With the above configuration, MASS generates a number of simulated attack sequences, some successful and others not. Table II shows the ground truth actions for one of the successful attack sequences. The attacker's first plan of action was to pass through Router 1 and compromise OSX1 using a DoS attack after first performing a network scan. However, MySQL was not running on OSX1, so the attacker was forced to continue compromising machines until a machine with MySQL was found. In this case, the attacker chose to use OSX1 as a stepping stone to gain access to the Windows subnet where numerous DoS attacks were performed on the subnet. This led to one of the machines to be compromised, where the mySQL server was found through Routers 4 and 5. The mySQL server was successfully attacked, using a DoS exploit, causing the goal state to be triggered.

Tables III and IV show sample outputs from a Snort sensor and a Reference sensor placed on Router 4. The Reference sensor is created to demonstrate the output of an ideal sensor with the ability to detect all suspicious activity except zero-day attacks. Since Snort does not provide an alert for every CVE, it is possible that an entire attack sequence could be executed using known CVE's without triggering any Snort alerts. Therefore, it may be useful in some cases to compare the Snort output with the Reference Sensor output to see which parts, if any, of the attack would have passed through the network undetected. In this case, only four Snort alerts were triggered on Router 4 even though the Reference sensor output shows that nine CVE's were exploited (some due to noise).

IV. Conclusion

The MASS system combines the context models representing the knowledge-of-us (VT.2 and VH) and knowledge-of-them (ABM and SGT), and produces a variety of attack sequences representing possible multistage attacks that could happen on an enterprise network. Data produced by MASS will benefit siginifciantly the Cyber SA community in analyzing the effectiveness of SA tools when facing diverse attack behaviors on different network configurations. Our limited sample results, due to the page limits, demonstrate the capability in producing realistic multistage attack sequences and the corresponding sensor observables.

Acknowledgement: This work is partially supported by US AFRL in collaboration with CUBRC. Approved for public release XX XX, 2014, #NNNNNNNN.

References

[1] A. Stotz and M. Sudit, "INformation fusion engine for real-time decision-making (INFERD): A perceptual system for cyber attack tracking," in *Proceedings of 10th International Conference on Information Fusion*, July 2007.

[2] S. Jajodia and S. Noel, "Advanced cyber attack modeling, analysis, and visualization," George Mason University, Tech. Rep., March 2010.

[3] S. Strapp and S. J. Yang, "Segmentating large-scale cyber attacks for online behavior model generation," in *Proceedings of International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, Washington, DC, April 1-4 2014.

[4] X. Qin and W. Lee, "Attack plan recognition and prediction using causal networks," in *Proceedings of 20th Annual Computer Security Applications Conference*. IEEE, December 2004, pp. 370–379.

[5] D. S. Fava, S. R. Byers, and S. J. Yang, "Projecting cyberattacks through variable-length markov models," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 359–369, September 2008.

[6] H. Du, D. F. Liu, J. Holsopple, and S. J. Yang, "Toward Ensemble Characterization and Projection of Multistage Cyber Attacks," in *Proceedings of the 19th International Conference on Computer Communications and Networks (ICCCN)*. Zurich, Switzerland: IEEE, August 2-5 2010.

[7] F. Cohen, "Simulating cyber attacks, defences, and consequences," *Computers & Security*, vol. 18, no. 6, pp. 479–518, 1999.

[8] ——, "Information system defences: A preliminary classification scheme," *Computers & Security*, vol. 16, no. 2, pp. 94–114, 1997.

[9] J. S. Park, J.-S. Lee, H. K. Kim, J.-R. Jeong, D.-B. Yeom, and S.-D. Chi, "Secusim: A tool for the cyber-attack simulation," in *Information and Communications Security*. Springer, 2001, pp. 471–475.

[10] I. Kotenko and E. Mankov, "Experiments with simulation of attacks against computer networks," in *Computer Network Security*, ser. Lecture Notes in Computer Science, V. Gorodetsky, L. Popyack, and V. Skormin, Eds. Springer Berlin Heidelberg, 2003, vol. 2776, pp. 183–194.

[11] N. Santhi, G. Yan, and S. Eidenbenz, "Cybersim: Geographic, temporal, and organizational dynamics of malware propagation," in *Proceedings of the Winter Simulation Conference*, ser. WSC '10. Baltimore, Maryland: Winter Simulation Conference, 2010, pp. 2876–2887.

[12] J. Holsopple, S. J. Yang, and B. Argauer, "Virtual terrain: a security-based representation of a computer network," in *Proceedings of SPIE Defense and Security Suymposium, Conference of Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security*, vol. 6973. Orlando FL, USA: SPIE, 2008. [Online]. Available: http://link.aip.org/link/?PSI/6973/69730E/1

[13] M. E. Kuhl, J. Kistner, K. Costantini, and M. Sudit, "Cyber attack modeling and simulation for network security analysis," in *Proceedings of the 39th Conference on Winter Simulation*. IEEE Press, 2007, pp. 1180–1188.

[14] S. Moskal, D. Kreider, L. Hays, B. Wheeler, S. J. Yang, and M. Kuhl, "Simulating attack behaviors in enterprise networks," in *Proceedings of IEEE Communications and Network Security*, Washington, DC, 2013.

[15] P. Xie, J. H. Li, X. Ou, P. Liu, and R. Levy, "Using bayesian networks for cyber security analysis," in *Proceedings of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2010, pp. 211–220.

[16] J. Holsopple and S. J. Yang, "FuSIA: Future situation and impact awareness," in *Proceedings of ISIF/IEEE International Conference on Information Fusion*, Cologne, Germany, July 2008.

[17] Z. Zhang, F. Naït-Abdesselam, and P.-H. Ho, "Boosting markov reward models for probabilistic security evaluation by characterizing behaviors of attacker and defender," in *Proceedings of Third International Conference on Availability, Reliability and Security*, March 2008, pp. 352–359.

[18] B. Wang, J. Cai, S. Zhang, and J. Li, "A network security assessment model based on attack-defense game theory," in *Proceedings of International Conference on Computer Application and System Modeling*, vol. 3, Oct 2010, pp. V3.639–V3.643.

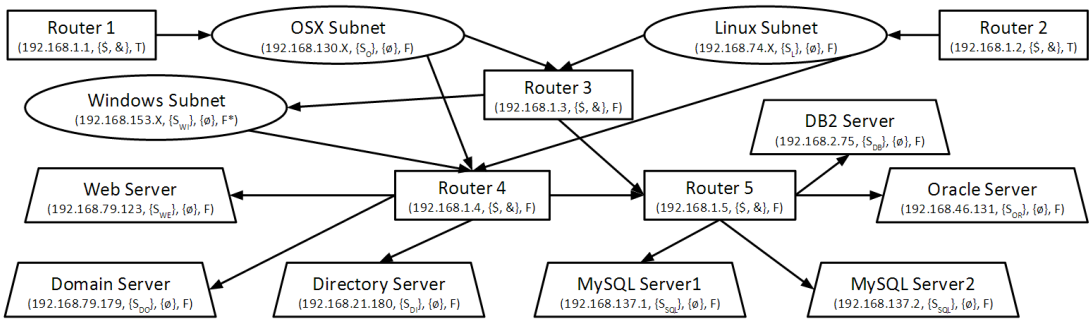

Fig. 5: Graphical representation of the target network

TABLE II: The ground truth output from the simulator

Time	Attack Sequence, SGT Stage, Attack Step	Current IP ->Target IP	Attack Description
01/17-11:04:25.527975:	Attack 1, Stage 1, Step 0 SUCCESSFUL:	92.163.255.56 ->192.168.130.1	(Reconnaissance->Network Scan->Generic->17320)
01/17-11:09:24.513657:	Attack 1, Stage 1, Step 1 SUCCESSFUL:	92.163.255.56 ->192.168.130.1	(DoS: resource consumption (memory)->119->apple:macosx->CVE-2013-0890)
01/17-11:23:40.469793:	Attack 1, Stage 3, Step 2 SUCCESSFUL:	192.168.130.1 ->192.168.153.10	(Reconnaissance->Network Scan->Generic->17320)
01/17-11:27:29.736260:	Attack 1, Stage 2, Step 3 UNSUCCESSFUL:	192.168.130.1 ->192.168.153.10	(DoS: crash / exit / restart->78->microsoft:windows2003server::sp2->CVE-2010-1885)
01/17-11:36:34.976108:	Attack 1, Stage 2, Step 4 SUCCESSFUL:	192.168.130.1 ->192.168.153.1	(Reconnaissance->Network Scan->Generic->17320)
01/17-11:38:44.195183:	Attack 1, Stage 2, Step 5 SUCCESSFUL:	192.168.130.1 ->192.168.153.1	(DoS: resource consumption (memory)->20->microsoft:windows dataaccess components:6.0->CVE-2011-0027)
01/17-11:45:51.970481:	Attack 1, Stage 3, Step 6 SUCCESSFUL:	192.168.130.1 ->192.168.153.5	(Reconnaissance->Network Scan->Generic->17320)
01/17-11:47:19.848965:	Attack 1, Stage 2, Step 7 SUCCESSFUL:	192.168.130.1 ->192.168.153.5	(DoS: resource consumption (memory)->20->microsoft:windows dataaccess components:6.0->CVE-2011-0027)
01/17-11:51:55.594684:	Attack 1, Stage 3, Step 8 SUCCESSFUL:	192.168.130.1 ->192.168.153.6	(Reconnaissance->Network Scan->Generic->17320)
01/17-11:58:57.176309:	Attack 1, Stage 2, Step 9 SUCCESSFUL:	192.168.130.1 ->192.168.153.6	(DoS: instability->362->microsoft:windowsserverr2008:r2:sp1:x64->CVE-2013-1269)
01/17-12:02:10.541085:	Attack 1, Stage 3, Step 10 SUCCESSFUL:	192.168.130.1 ->192.168.153.2	(Reconnaissance->Network Scan->Generic->17320)
01/17-12:02:43.297270:	Attack 1, Stage 2, Step 11 SUCCESSFUL:	192.168.130.1 ->192.168.153.2	(DoS: resource consumption (other)->362->microsoft windowsserver 2008:r2:sp1:x64->CVE-2013-1252)
01/17-12:09:38.928465:	Attack 1, Stage 2, Step 12 UNSUCCESSFUL:	192.168.130.1 ->192.168.153.4	(DoS: crash / exit / restart->119->microsoft:windows dataaccess components:6.0->CVE-2012-1891)
01/17-12:13:10.686726:	Attack 1, Stage 2, Step 13 SUCCESSFUL:	192.168.130.1 ->192.168.153.4	(DoS: crash / exit / restart->119->microsoft:windows dataaccess components:6.0->CVE-2012-1891)
01/17-12:56:48.941972:	Attack 1, Stage 3, Step 14 SUCCESSFUL:	192.168.153.4 ->192.168.137.1	(Reconnaissance->Network Scan->Generic->17320)
01/17-13:19:13.241854:	Attack 1, Stage 3, Step 15 SUCCESSFUL:	192.168.153.4 ->192.168.137.1	(DoS: resource consumption (CPU)->89->oracle:mysql:5.5.12->CVE-2012-4414)

TABLE III: Snort Sensor Output- Router 4

Time/Snort ID		Signature	Classification	Priority	Protocol	Current IP ->Target IP
01/17-10:57:38.840727 [1:24766:2]	[**]	SERVER-WEBAPP Novell File Reporter SRS request arbitrary file download attempt [**]	[Classification: Attempted Administrator Privilege Gain]	[1]	tcp	192.168.153.3: ->192.168.1.4:18580
01/17-11:12:40.055603 [1:18973:3]	[**]	BROWSER-WEBKIT Apple Safari Webkit button first-letter style rendering code execution attempt [**]	[Classification: Attempted User Privilege Gain]	[1]	tcp	192.168.153.3: ->192.168.1.4:47806
01/17-11:34:01.578360 [1:19943:6]	[**]	FILE-OFFICE Microsoft Office Excel MsoDrawingGroup record remote code execution attempt [**]	[Classification: Attempted User Privilege Gain]	[1]	tcp	192.168.2.75: ->192.168.1.4:759
01/17-11:52:38.355930 [1:20249:4]	[**]	SERVER-OTHER Java Web Start BasicService arbitrary command execution attempt [**]	[Classification: Attempted User Privilege Gain]	[1]	tcp	192.168.74.5: ->192.168.1.4:42847

TABLE IV: Reference Sensor Output- Router 4

Time	Current IP ->Target IP	Attack Description
01/17-10:57:38.840727	192.168.153.3 ->192.168.1.4	Absolute path traversal vulnerability in NFRAgent.exe in Novell File Reporter allows attackers to read arbitrary files
01/17-11:11:11.386424	192.168.153.5 ->192.168.1.4	Buffer overflow in Microsoft PowerPoint 2002 SP3 and 2003 SP3 allows remote attackers to execute arbitrary code..."
01/17-11:12:40.055603	192.168.153.3 ->192.168.1.4	Use-after-free vulnerability in WebKit in Apple Safari before 5.0 on Mac OS X 10.5 through 10.6...
01/17-11:34:01.578360	192.168.2.75 ->192.168.1.4	Stack-based buffer overflow in Excel and Office XP SP3 allows attackers to execute arbitrary code...
01/17-11:36:09.065227	192.168.130.1 ->192.168.1.4	Multiple unspecified vulnerabilities in WorldClient in Alt-N MDaemon before 10.02 have unknown impact...
01/17-11:52:38.355930	192.168.74.5 ->192.168.1.4	The BasicService in Sun Java Web Start allows remote attackers to execute arbitrary programs on a client machine...
01/17-12:06:15.993494	192.168.74.2 ->192.168.1.4	Open redirect vulnerability in phpgwapi/ntlm/index.php in EGroupware Enterprise Line (EPL) before 11.1.20110804-...
01/17-12:57:30.045452	192.168.130.1 ->192.168.1.4	Use-after-free vulnerability in mm/mprotect.c in the Linux kernel before 2.6.37-rc2 cause a denial of service...
01/17-13:19:13.241854	192.168.153.4 ->192.168.137.1	Multiple SQL injection vulnerabilities in the replication code in Oracle MySQL possibly before 5.5.29, and MariaDB 5.1.x through 5.1.62, 5.2.x through 5.2.12, 5.3.x through 5.3.7, and 5.5.x through 5.5.25, allow remote authenticated users...