Terrain and Behavior Modeling for Projecting Multistage Cyber Attacks

Daniel Fava† Jared Holsopple‡ Shanchieh Jay Yang† Brian Argauer†

† Department of Computer Engineering
Rochester Institute of Technology
Rochester, New York, U.S.A

‡ Information Fusion Division
Calspan-UB Research Center
Buffalo, New York, U.S.A

Abstract— **Contributions from the information fusion community have enabled comprehensible traces of intrusion alerts occurring on computer networks. Traced or tracked cyber attacks are the bases for threat projection in this work. Due to its complexity, we separate threat projection into two subtasks: predicting likely next targets and predicting attacker behavior. A virtual cyber terrain is proposed for identifying likely targets. Overlaying traced alerts onto the cyber terrain reveals exposed vulnerabilities, services, and hosts. Meanwhile, a novel attempt to extract cyber attack behavior is discussed. Leveraging traditional work on prediction and compression, this work identifies behavior patterns from traced cyber attack data. The extracted behavior patterns are expected to further refine projections deduced from the cyber terrain.**

Keywords: prediction, cyber security, contextual reasoning

I. Introduction

The task of assessing cyber attacks has drawn increasing attention from the information fusion community. Drawing analogies from traditional fusion problems, assessing cyber attacks involves detecting, tracking, correlating, and projecting attack movements. Malicious activity on computer networks triggers Intrusion Detection Systems (IDSs) to produce alerts. Each attack action may trigger zero, one, or many alert messages. Correlating and filtering alert messages, *i.e.,* observables, provide traces or tracks of ongoing multistage attacks in a computer network. Precise and timely predictions shall lead to better decision making and minimal operational interruptions when combating cyber attacks. Therefore, the focus of this paper is on projecting the movements of tracked cyber attacks.

Projecting cyber attack actions depends on the detection and tracking of malicious activity. Host-based and network-based IDSs typically monitor application and operating system level activity as well as network traffic. Alerts are generated when monitored activity matches one or more signatures of previously known attacks (signature-based) or is abnormal and suspicious (anomaly-based). Non-intrusive and intrusive malicious activity detection has been widely tackled yet still continuously poses challenges, due to the constantly evolving nature of vulnerabilities and, consequently, changes in exploitation mechanisms. As a result, alert messages produced by IDSs may be incomplete and misleading.

In 2000, Bass [1] advocated the need of information fusion when facing overwhelming number of alerts reported on typical enterprise networks. Since then, much work has been devoted to alert correlation, *e.g.,* [2]–[10]. Correlating IDS alerts involves reasoning based on, primarily, the source and target IP addresses, the attack type descriptions from the alert messages, and the time interval between alerts. This set of information helps revealing the courses of action potentially taken by multistage attacks, which may span over multiple machines or subnets. Alerts that belong to the same course of action are grouped and traced to form an attack track. Each attack track, which may be modeled as a directed graph, illustrates the causal and sequential relationships between alerts belonging to the same multistage attack. Note that undetected activity and excessive alerts (typically due to reconnaissance activity) may lead to mis-correlated alerts or fragmented attack tracks.

While alert correlation is still under investigation for better accuracy and real-time operation, the next challenge is to project attack actions based on the detected attack tracks. Cyber attack projection may be categorized as a L3 fusion problem based on the Joint Directors of Laboratories (JDL) fusion model [11], [12] and its revision in 2004 [13]. The challenge of projecting attack actions comes from the fact that cyber attacks are diverse and constantly changing in terms of not only intents and exploitation methods, but also network and system configurations. Common approaches in assessing courses of actions (not necessary for cyber attacks) include Bayesian Network and Hidden Markov Models. In the cyber domain, much work has been focused on assessing the sequence over which system vulnerabilities may be exploited in a network, *e.g.,* [14]–[16]. Qin and Lee [17] proposed to adapt attack plan model using Bayesian Network as attack actions are detected. Though theoretically sound, one problem in these approaches is the complexity involved in developing and/or maintaining the models for courses of actions, which encompass the diverse exploitation methods, intents, and network configurations. Holsopple, Yang, and Sudit [18] proposed to simplify the problem by separating the modeling of network configuration and cyber attack methods. As attack actions are detected, assessments are performed independently using the two predetermined models. The assessments are then fused to determine the targeted entities (intents of the attacks) in the network.

This work extends the concept of separating the modeling of network and system configuration from the extraction of attack behavior. In Section II, we investigate which critical information of a computer network is necessary for threat prediction, and whether this information can be obtained and updated automatically. We propose a virtual cyber terrain that models the accessibility or exposure of system vulnerabilities at different network access domains. The cyber terrain model is not goal-oriented like a typical vulnerability tree, nor does it utilize probabilities such as a Bayesian network. It is a directed graph containing critical topological and system configuration information for situation and threat assessment caused by cyber attacks.

In Section III, we attempt to extract patterns from traced cyber attack actions. Note that behavior is influenced by the attacker's intent, his or her preferred exploit sequences and capabilities, and the network and system vulnerability exposed to the attacker. General perception is that cyber attack behavior can be diverse and changing. We conduct experiments using traced ground truth data that do not contain exploit evolution, *i.e.,* there is a finite set of attack types. We leverage traditional work on prediction, which has a significant overlap with the study of data compression [19]. A customized suffix tree is developed to examine what trends, if any, may exist in the types of attacks an attacker may choose to execute.

Cyber terrain, which determines vulnerabilities and possibly exposed targets, along with behavior extraction may provide efficient and accurate projection of cyber threats. The next two sections illustrate the proposed cyber terrain and our findings on the behavior extraction experiments.

II. Contextual Reasoning via Cyber Terrain Modeling

A reasonably secured network typically has multiple access domains, where direct access to internal and often critical domains or subnets is prohibited. Serious cyber attacks, thus, need to exploit different system vulnerabilities and progress through multiple domains. Reasoning on the progress made by a cyber attack shall benefit from a contextual model - a virtual cyber terrain that models the logical accessibility from one access domain to another. Most importantly, the cyber terrain should model the system and network configurations, including their vulnerabilities that may be exposed as the attacker compromised one or more systems in the network.

Vidalis and Jones [15] proposed the use of a vulnerability tree to identify the types of attacks an attacker could perform to accomplish a goal. Their model requires a separate vulnerability tree for each possible goal, which could be potentially numerous. Philips and Swiler [14] and Liu and Man [16] suggested the use of a Bayesian network to model the vulnerabilities. Their model assumes acyclic graphs, which implies that bi-directional connections between hosts must be modeled in separate acyclic graphs. Massicote *et. al.* [20] discussed ways of introducing contextual information to cross examine with reported IDS alerts and, thus, reduce false positives. Their experiences suggested that contextual information may be derived by utilizing Snort [21], Nessus [22] and Bugtraq [23]. Our model, developed independently of Massicote's work, shares some similar ideas, yet provides additional network connectivity and privilege information for situational assessment and threat prediction.

A. Cyber Terrain Definition

We model a cyber terrain as a directed graph consisting of host or cluster nodes that are interconnected with directed arcs. Both the nodes and arcs have attributes defined for threat assessment and projection. Each node in the cyber terrain contains the following attributes:

- A node identifier
- IP address(es) (and host name(s) if applicable)
- Service and data criticality metrics
- Service Tree(s)

Note that a single node may have multiple IP addresses to define a cluster of identically configured hosts or servers. This simplifies the terrain model as well as the impact assessment process. The service and data criticality metrics are numerical numbers between 0 and 1, defining the importance of services provided by the node and data content stored in the node, respectively. A value of 0 means that the node is irrelevant, whereas a value of 1 corresponds to a host or cluster that contains most valuable content or most critical to network operation. Note that both criticality measures may be defined by the network analyst and/or referencing to databases such as CVE [24]. A thorough study of how criticality may be defined to better assess the consequences of cyber attacks is needed. This work focuses on 'projecting' cyber attack actions and assumes that the numerical values are given.

Each node may have one or more services running; here we define service in a general term that includes both remote services and local user applications. A *service tree* is used to represent each service available in the node. At the terrain model development phase, all running services will be scanned by tools to determine the open ports and vulnerabilities of each machine. This information then can be used to build the service trees in each machine. To build these trees, one or more databases must be cross-referenced to determine which exposures and exploits should be mapped to each version of the running services. As also suggested by Massicote *et. al.* [20], we adopt Snort [21], Bugtraq [23], Nessus [22], and NMAP [25] to build a *service database* that will aid the creation of a service tree. The service tree will capture the name of the service, versions of that service that are running, and the IDS alerts that could be reported if the corresponding vulnerabilities were exploited. An example template of a service tree is shown in Figure 1.

A key feature of the service tree model presented here is that it captures privilege differences between services and between versions of the same service. Like regular users, every service runs at a given privilege level. While many services do run at the system level, other services can only run at the user level. If a service is exploited, the attacker usually gains access to the computer at the level of the service. The service

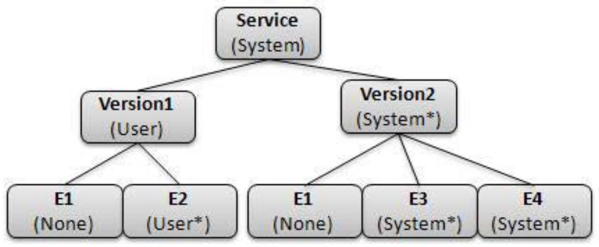

Fig. 1. An example service tree template with the privilege shown in parentheses, where an asterisk (*) implies that the privilege was inherited from the parent.

database contains a default privilege for every service. Should the administrator change the privilege of the service, the terrain will be updated accordingly. In addition, services may be local or remote. A remote service can be possibly compromised without obtaining access privilege to the computer hosting the service. Such services typically listen on a specified TCP/UDP port. A local service is a service that can only be exploited after the attacker gains access to the computer hosting the service.

The version(s) of each service are on the second level of the tree. It is possible that multiple versions of the same service could be running on the same host. For example, a web programmer may wish to run multiple versions of Mozilla Firefox concurrently to test for the compatibility of a web site between different versions. Some vulnerabilities could be fixed or introduced from version to version. Also, a specific version may also have a privilege different from that of its parent service. Therefore, our model allows the privilege to be defined explicitly for each version of each service. By default, if no privilege is defined, a version will inherit the privilege of its parent service.

The actual IDS alerts are the children of the versions. The IDS alerts classified under a service and version imply that if an IDS alert was reported on that host, the parent version of the service is affected by that alert. Like versions, these IDS alerts can also inherit the privilege level of its parent. Some IDS alerts may not correspond to actual vulnerability exploits. For example, knowledge discovery attack such as a TCP Syn attack may indicate that the target host is alive, but will not compromise any privileges to the host. Such alerts are defined with a 'None' privilege.

The service trees provide a structural model to determine the extent to which services are compromised on each host. More specifically, it helps to determine the privilege(s) obtained by the attacker during the process of an attack. It also filters out false positives, *i.e.,* alerts that do not correspond to a service running on the target host or subnet. Furthermore, and perhaps more importantly, by correlating the services and privileges in different machines, the cyber terrain may be used to deduce potentially threatened targets with similar or the same running services. Inference using the services, however, depends on the connectivity allowed between hosts and subnets. This connectivity is defined by the arcs in the cyber terrain model. A simplified example of a cyber terrain model is shown in Figure 2 for illustration purposes. The model has only three nodes interconnected with attributed directed arcs. A more realistic model may contain many more nodes and each service tree may be associated with tens or hundreds of IDS alerts.

In analyzing the progression of cyber attacks, the physical topology of the network is not entirely relevant. Routers and switches allow communications between hosts despite them not being physically connected. The communications between hosts, however, are subject to the configuration of the hosts themselves as well as the configuration of the routers and switches between the hosts. The attributes of the directed arcs in the cyber terrain are defined to capture these restrictions. Two attributes are defined with every arc: Banned and Access List. The Access List contains a list of protocols and IP addresses. The Banned attribute is a Boolean value where a value of 'false' implies that the protocols and IP addresses on the Access List are the only protocols and addresses allowed across the directed arc, and everything else is blocked. A value of 'true' defines the opposite.

B. Generating a Cyber Terrain

The creation of a cyber terrain involves determining services running on each machine and cross-referencing those services with relevant vulnerabilities. Given the large variation in services that could run on a host and the large number of IDS alerts corresponding to vulnerabilities of the services, it is not realistic to manually create an accurate, complete cyber terrain for even a small network.

To automate the creation of a cyber terrain, a database mapping IDS alerts to susceptible services and versions is necessary. From the IDS alert alone, it is impossible to accurately determine what service was compromised by that attack. However, databases, *e.g.,* Nessus and Bugtraq, provide information on which services (and versions) are susceptible to which vulnerabilities. Massicote *et. al.* [20] noted that 47% of Snort alerts did not provide Nessus or Bugtraq references, so those alerts need to be manually classified. This, however, would only be a one-time inconvenience.

Scanners such as Nessus and NMAP can be used to scan the network for the remote services running on each machine. Once the services are identified, the database discussed above can be queried for relevant IDS alerts. The service trees then can be created for each host. The network scanning provides only remotely exploitable services. Local services would need to be identified by the administrator or a local scan of each host.

The directed arcs representing allowed and banned protocol communication between hosts are critical to the terrain model. One possible way to define the arcs is by having each host scan all other hosts to determine remote access protocol or ports allowed or banned. This could, however, generate unwanted traffic. Another method is to analyze router and firewall configurations and determine the set of protocols allowed or banned between access domains or subnets.

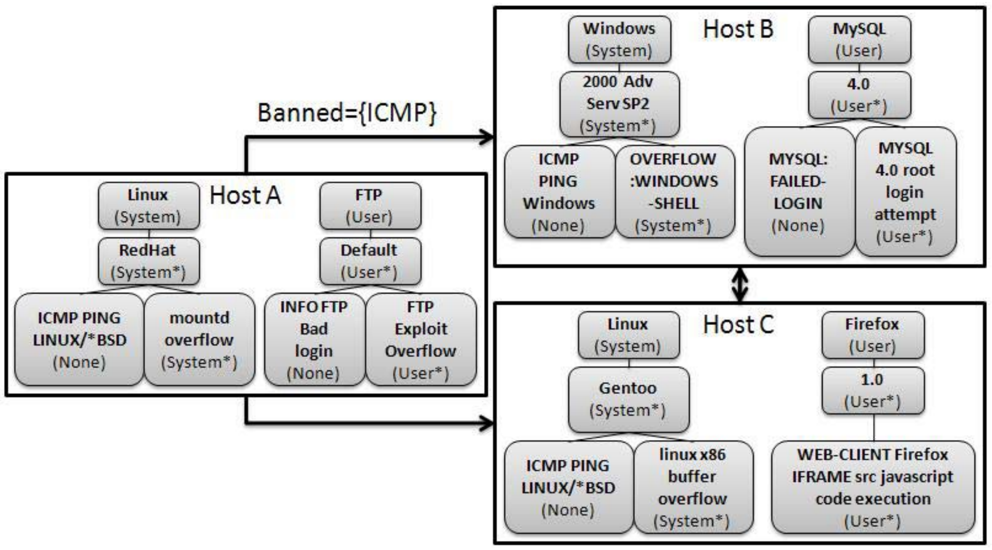
Banned={ICMP}
Host A
Linux (System)
RedHat (System*)
ICMP PING LINUX/*BSD (None)
mountd overflow (System*)
FTP (User)
Default (User*)
INFO FTP Bad login (None)
FTP Exploit Overflow (User*)
Host B
Windows (System)
2000 Adv Serv SP2 (System*)
ICMP PING Windows (None)
OVERFLOW :WINDOWS -SHELL (System*)
MySQL (User)
4.0 (User*)
MYSQL: FAILED-LOGIN (None)
MYSQL 4.0 root login attempt (User*)
Host C
Linux (System)
Gentoo (System*)
ICMP PING LINUX/*BSD (None)
linux x86 buffer overflow (System*)
Firefox (User)
1.0 (User*)
WEB-CLIENT Firefox IFRAME src javascript code execution (User*)

Fig. 2. An example of cyber terrain model where three nodes are connected with logical arcs.

C. Threat Projection using Cyber Terrain

The cyber terrain model is designed to capture the relationship between services and privileges associated with the hosts. As IDS alerts are grouped into attack tracks, a threat assessment may use a cyber terrain model to determine which services are compromised, to what extent, and, then, which services are at a higher risk given the current situation. We shall illustrate the use of cyber terrain using the example shown in Figure 2. Consider the following four-alert attack track.

1) 'ICMP PING LINUX/*BSD' (Internet → Host A, TCP)
2) 'Linux x86 buffer overflow' (Internet → Host A, TCP)
3) 'Mountd overflow' (Internet → Host A, TCP)
4) 'ICMP PING Windows' (Host A → Host B, ICMP)

The first alert 'ICMP PING LINUX/*BSD' is mapped to a leaf of the RedHat Linux service tree in Host A with a privilege of 'None'. The cyber terrain model reveals that the attacker knows Linux is running on Host A, yet he gains no privileges to Host A because it is just a knowledge discovery attack. The attacker then executes a 'Linux x86 buffer overflow' attack with the knowledge that Host A is running Linux. However, the Linux version running on Host A is not vulnerable to such an attack. This is deduced since 'Linux x86 buffer overflow' is not a child alert in the Host A model. The second attack attempt is considered a failure. The attacker then tries again with the *Mountd overflow* attack. Since this is an alert child node in Host A, it can be assumed that Host A is now compromised with the attacker gaining system-level privilege. The most recent action detected is an 'ICMP PING Windows' action for Host B. Although Host B is a Windows machine, this is also a failed attempt because the arc between Hosts A and B does not permit ICMP traffic.

Both the success and failed attempts are indicative to potential future attack actions. The use of cyber terrain provides contextual reasoning in terms of 'exposed vulnerabilities', 'demonstrated capabilities', and 'host criticality'.

- **Demonstrated Capabilities**: Despite that failed attacks do not pose any threat to the integrity of the network, they do provide information for threat assessment. Failed attacks give a clue to the capability of the attacker. In the example above, 'Linux x86 buffer overflow' failed for Host A; yet, it would have been successful had the attacker executed it on Host C. Since there is no banned protocol between Host A and C and Host C is also susceptible to executed ICMP PING LINUX/*BSD', Host C is likely to be a victim of next attack. Hosts that have the same services or similar services may be susceptible to a common set of attacks. The cyber terrain model allows identifying the same alert nodes in different machines that are connected with no conflicting banned protocol.
- **Exposed Vulnerabilities**: In many cases, an attacker discovers the running services on a machine and then decides on which exploits to execute. The example above demonstrates that the attacker attempted two Linux-based intrusive exploits, one failed and the other succeeded. A cyber terrain model in real life will have many service trees for each host and many alert nodes for each service tree. Thus, a reasoning based on which service vulnerabilities are discovered gives indication of the type of exploits the attacker may attempt next.
- **Host Criticality**: The numerical criticality values present two uses for threat assessment. First, a host that contains critical data or essential to a network's operation could be more likely to be the target of an attack. Second, as the criticality may be derived based on traffic volume, a host with high criticality may lead to more machines exposed to the attacker (*i.e.,* connected with no banned protocol).

In this case, a higher weight should be placed on the compromised hosts that have higher criticality score when applying the previous two rules.

III. Cyber Attack Behavior Extraction and Modeling

With cyber terrain providing projections of targets exposed to detected attacks, this section discusses our approach to predict based on attacker's behavior or tendency. To our knowledge, no similar work has been conducted. In fact, prior to this investigation, we were unsure whether there existed behavior patterns to be extracted.

We consider IDS alerts as observables of tracked attack actions. These tracks, however, may be filled with 'noise' - one attack action may trigger multiple alerts from one or more IDSs. Thus, we filter by removing alerts that have the same description fields (signature, source IP, target IP, etc.) as the preceding alert, and if they fall within $\Delta t = 1$ second from each other. Note that our heuristic filtering process does not re-create the actual attack actions; however, we believe that an analyst or an expert system may be able to extrapolate behavior from these filtered alert sequences. The resulting sequences are the basis of our behavior prediction process.

Each filtered attack sequence consisting of n alerts is converted to $s = \{x_1, x_2, ...x_n\}$ where x_i belongs to the alphabet Ω. The choice of field or fields for characterizing the cyber attack will have a direct impact on Ω. For example, the larger the pool of alert fields used, the larger the alphabet size will be. This work considers the description field from Snort [21] alerts. This choice is made to reflect the attacker's tendency in choosing reconnaissance and exploitation methods, and allows mapping to system and network vulnerabilities.

A. Finite-context models and Suffix Trees

Leveraging the work on compression and prediction [19], we adopt a context-based model in which future events of an n^{th} order model depend on the n previous observations:

$$P^n\{x_{t+1}|x_{t-n+1}, ..., x_t\}. \tag{1}$$

A modified suffix tree is implemented to record all suffixes of the filtered attack sequences. The algorithm was motivated by the work of Begleiter, El-Yaniv, and Yona [26], and modified to take a set of finite length sequences instead of a single long sequence of observations. To illustrate, a tree built based on a single sequence '$+FGGFGF*$' is shown in Figure 3. Note that 'F' and 'G' correspond to Snort alerts 'WEB-IIS nsiislog.dll access' and 'WEB-MISC Invalid HTTP Version String' and '$+$' and '$*$' mark the start and the end of the attack sequence. Edges are weighed with the number of times the suffix tree is traversed through that branch. For examples, '$FGF*$' happened only once in the sequence while 'FG' happened twice.

A suffix tree can be used for prediction once its been built from representative attack sequences. Prediction involves determining the most likely future action (x_{t+1}) given an

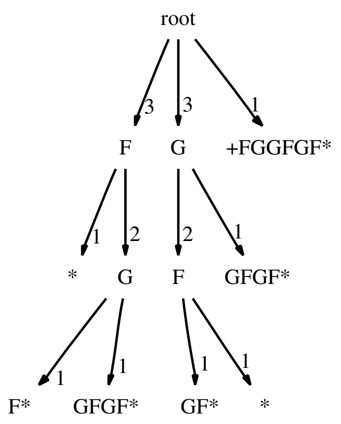

Fig. 3. The suffix tree for a finite sequence '$+FGGFGF*$'.

unfolding sequence of alerts $s = \{x_1, x_2, ..., x_t\}$. The n^{th} order prediction may be expressed as follows.

$$x_{t+1} = \arg\max_{x_i \in \Omega} P^n\{x_i|x_{t-n+1}, ..., x_t\}, \tag{2}$$

where $n \leq t$. The probability P^n is derived from the suffix tree branch counts and it reflects the probability of an n^{th} finite-context model, that is, n previous observations are taken into account when computing P^n.

For example, consider using the suffix tree shown in Figure 3 to predict the next attack action following '$+GF$'. The minus one order prediction will give an equal probability to all possible values in the alphabet Ω. The zero order prediction will conclude that G and F are equally likely with $P^0\{F\} = P^0\{G\} = 3/7$ (with $P^0\{*\} = 1/7$. The first order model will predict G since $P^1\{G|F\} = 2/3$ and $P^1\{*|F\} = 1/3$. The second order model will predict that G and $*$ are equally likely with $P^2\{G|GF\} = P^2\{*|GF\} = 1/2$.

A logical extension of the n^{th} order prediction models is to *combine* or *blend* probabilities from several finite-context predictors. Let $P^o(x)$ be the probability of a character $x \in \Omega$ happening according to the o^{th} order model given a suffix tree and an observed sequence. The blended probability is the weighted sum of $P^o(x)$:

$$P(x) = \sum_{o=-1}^{m} w_o \times P^o(x) \tag{3}$$

where m is the longest match for the observed sequence s in the suffix tree.

Probabilities for a 0^{th} order model (P^0) are the normalized counts of all possible characters in the alphabet. A minus one order model is used such that $P^{-1}(x) = 1/|\Omega|$ for all $x \in \Omega$. Interested readers may refer to [19] for a detailed discussion on the minus one order model and the *zero frequency problem*.

The weights w_o can be derived with different blending schemes. This work adopts a blending scheme from [19]:

$$\begin{aligned} w_m &= (1 - e_m) \\ w_o &= (1 - e_o) \times \prod_{i=o+1}^{m} e_i, \; -1 \le o < m \\ e_o &= \frac{1}{c_o + 1} \end{aligned}$$

where c_o is the number of length-o sequences in the suffix tree. The blended predictor will be referred to as the Variable Length Markov Model (VLMM).

B. Experiments and Discussion

We conduct a series of experiments using the cyber attack data generated by Skaion Corporation on a VMware network with actual scripted attacks [27]. The data set includes a total of 1,113 attack sequences and a total of 4,756 alerts after the filtering process. The data set contains 47 distinct attack types that form the alphabet Ω. The length of the filtered attack sequence varies from 2 to 220. The goal is to determine whether attack patterns exist and are indicative for future exploit methods. The attack data set is randomly split into two halves, one for building the suffix tree and the other for testing. Ten independent tests with different random splits of the data set were used for the results reported in this section.

Predictions based on the 0^{th}, 1^{st}, 2^{nd}, and 3^{rd} order finite-context models as well as predictions based on the VLMM were compared against ground truth. The performance was measured by the prediction rate (%), which is defined as the number of predictions that are correct over the total number of predictions. Prediction rates of each model are calculated with respect to the top candidate (the one with the highest probability), with the top two candidates, and with the top three candidates.

Figure 4 shows the top-1, top-2, and top-3 prediction rates for each model; each data point is averaged over 10 independent runs with error bars corresponding to ± 1 standard deviation. It can be clearly seen that the 0^{th} and the 4^{th} order models perform inferior to the 1^{st}, 2^{nd}, and VLMM. The poor

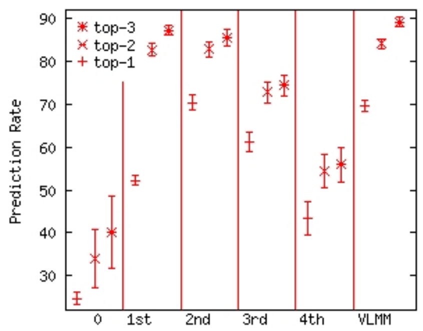

Fig. 4. The top-1, top-2, and top-3 prediction rates achieved by the various context-based models.

$\tilde{4}0\%$ top 3 prediction rate exhibited by the 0^{th} order model validates that the most frequent exploits are not at all indicative of attack behavior; consider that an experienced analyst may be able to deduce a few subsequent exploits after observing the attack sequence. The poor prediction rate (less than 50%) exhibited by the 4th order model suggests that inference with long contexts can be misleading. Higher-order models perform even worse. Studying the training and test sets, we found that long attack sequences are rare and by and large different from each other. Our results strongly suggest that the 1^{st} and the 2^{nd} order contexts are most useful in predicting cyber attack exploits. In other words, the next exploit has a strong correlation with the previous one or two exploits. The VLMM that blends the predictions from various orders takes such advantage and performs well with near 90% prediction rate given its top 3 predictions.

One noticeable characteristic of the data set is the 'homogeneity' of characters within an attack sequence. Our data set seems to be dominated by few characters that tend to repeatedly appear in the same sequence even after the filtering process. The top three attacks (out of 47) occurring in the entire data set are 'ICMP PING NMAP', 'WEB-MISC Invalid HTTP Version String', and '(http_inspect) BARE BYTE UNICODE ENCODING', which account for 43.5%, 22.4%, and 9.0%, respectively. In particular, 'ICMP PING NMAP' is followed by 'ICMP PING NMAP' 87.7% of the time (instead of followed by other attack actions). This leads to a question: is our prediction biased due to the repeating and excessive 'ICMP PING NMAP'?

An experiment was conducted with the prediction rates calculated for two cases: when consecutive attack descriptions are the same (repeating attacks) and when consecutive attack descriptions are different (non-repeating attacks). Figure 5 shows the top-3 prediction rates achieved by the different models for the repeating and non-repeating attacks. The results clearly suggest that repeating attacks are more accurately predicted, with over 95% prediction rates achieved by 1^{st}, 2^{nd}, and VLMM. The non-repeating attacks are not as accurately predicted. Only VLMM achieves over 80% prediction rate when considering the top-3 predictions.

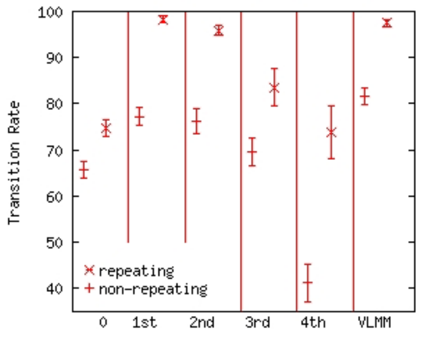

Fig. 5. The top-3 prediction rates achieved by the various context-based models when predicting repeating and non-repeating attacks.

Figure 4 would lead one to conclude that 1^{st} and 2^{nd} order models perform approximately equal to VLMM. However, Figure 5 shows evidence that blending higher order contexts does help improve prediction for non-repeating attacks. VLMM may significantly outperform 1^{st} and 2^{nd} order models for data sets characterized by non-repeating attacks. In addition, better blending or filtering schemes may yield even higher prediction rates for these types of data sets.

IV. Conclusion

This paper proposes two approaches that can be jointly used for threat projection of cyber attacks. The proposed virtual cyber terrain aims at capturing the relationships between hosts, services, and privileges in a computer network. Overlaying traced IDS alerts onto the cyber terrain allows contextual reasoning based on vulnerabilities exposed, service similarity, and the host criticality. The projected threats based on cyber terrain can be potentially reduced based on attack behavior inference. A general context-based model is adopted to extract patterns exhibited from traced attack data. Our experiments suggest that the past two attack actions are most indicative to the next attack action in the same attack track, and that longer contexts help if observed attack actions change frequently.

Many questions arise from this work. Automating the generation and updating of cyber terrain is a challenging task that may require the development of new network and host scanning tools. The definition of host criticality and its use should be further investigated. The cyber terrain model needs to be tested across different data sets to determine how effective the defined relationships are in projecting attacks. The novel attempt in developing a cyber attack behavior prediction model has shown promising results. However, there is a large room for improvement with a better filtering process and a better blending scheme. Ongoing work is to investigate a predictor that is immune to missing, redundant, and incorrect observations. Finally, this research would benefit from a justifiable mathematical definition to adjust cyber terrain projections based on the behavior models.

ACKNOWLEDGMENTS

This work is funded through the National Center for Multisource Information Fusion (NCMIF) grant under the technical supervision of AFRL/IFEA. The authors would like to thank Adam Stotz, Moises Sudit, John Salerno, Michael Hinman, and George Tadda for their comments toward the development of the virtual cyber terrain and providing the data set for behavior extraction.

References

[1] T. Bass, "Intrusion detection systems and multisensor data fusion," *Communications of the ACM*, vol. 43, no. 4, April 2000.

[2] A. Valdes and K. Skinner, "Probabilistic alert correlation," in *Recent Advances in Intrusion Detection (RAID 2001)*, ser. Lecture Notes in Computer Science, no. 2212. Springer-Verlag, 2001. [Online]. Available: http://www.sdl.sri.com/papers/raid2001-pac/

[3] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," in *Proceedings of 2002 IEEE Symposium on Security and Privacy*, 2002, pp. 202–215.

[4] P. Ning, Y. Cui, and D. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM Conference on Computer & Communications Security*, 2002, pp. 245–254.

[5] F. Valeur, G. Vigna, C. Kruegel, and R. A. Kemmerer, "Comprehensive approach to intrusion detection alert correlation," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 3, pp. 146–169, July-Sep 2004.

[6] P. Ning, Y. Cui, D. Reeves, and D. Xu, "Tools and techniques for analyzing intrusion alerts," *ACM Transactions on Information and System Security*, vol. 7, no. 2, pp. 273–318, May 2004.

[7] D. Xu and P. Ning, "Alert correlation through triggering events and common resources," in *Proceedings of 20th Annual Computer Security Applications Conference*, December 2004, pp. 360–369.

[8] M. Sudit, A. Stotz, and M. Holender, "Situational awareness of a coordinated cyber attack," in *Proceedings of SPIE, Defense and Security Symposium*, March 2005, pp. 114–129.

[9] S. T. King, Z. M. Mao, D. G. Lucchetti, and P. M. Chen, "Enriching intrusion alerts through multi-host causality," in *Proceedings of the 2005 Network and Distributed System Security Symposium (NDSS'05)*, February 2005.

[10] S. Mathew, D. Britt, R. Giomundo, S. Upadhyaya, M. Sudit, and A. Stotz, "Real-time multistage attack awareness through enhanced intrusion alert clustering," in *Proceedings of IEEE Military Communications Conference, MILCOM*, October 2005, pp. 1–6.

[11] U.S. Department of Defense, Data Fusion SubPanel of the Joint Directors of Laboratories, "Technical panel for C3," in *Data Fusion Lexicon*, October 1991.

[12] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," in *Proceedings of the IEEE*, vol. 85, no. 1, January 1997, pp. 6–23.

[13] J. Llinas, C. Bowman, G. Rogova, A. Steinberg, E. Waltz, and F. White, "Revisions and extensions to the JDL data fusion model II," in *Proceedings of The 7th International Conference on Information Fusion*, June 2004, pp. 1218–1230.

[14] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the 1998 workshop on New security paradigms*. New York, NY, USA: ACM Press, 1998, pp. 71–79. [Online]. Available: http://doi.acm.org/10.1145/310889.310919

[15] S. Vidalis and A. Jones, "Using vulnerability trees for decision making in threat assessment," University of Glamorgan, School of Computing, Tech. Rep. CS-03-2, June 2003.

[16] Y. Liu and H. Man, "Network vulnerability assessment using Bayesian networks," in *Proceedings of Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security*, vol. 5812, March 2005, pp. 61–71.

[17] X. Qin and W. Lee, "Attack plan recognition and prediction using causal networks," in *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04)*, December 2004.

[18] J. Holsopple, S. J. Yang, and M. Sudit, "TANDI: Threat assessment for networked data and information," in *Proceedings of SPIE, Defense and Security Symposium*, vol. 6242, April 2006.

[19] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text Compression*. Prentice Hall, 1990.

[20] F. Massicotte, M. Couture, L. Briand, and Y. Labiche, "Context-based intrusion detection using Snort, Nessus and Bugtraq databases," in *Proceedings of the Third Annual Conference on Privacy, Security and Trust, Fredericton*, October 2005.

[21] Sourcefire, "Snort: an open source network intrusion prevention and detection system," 2006, `http://www.snort.org`.

[22] Tenable Network Security, Inc., "Nessus vulnerability scanner," 2006, `http://www.nessus.org/`.

[23] SecurityFocus, "Bugtraq vulnerability database," 2006, `http://www.securityfocus.org/bid`.

[24] Mitre, "Common vulnerabilities and exposures (CVE dictionary)," 2007, `http://cve.mitre.org/`.

[25] Insecure.com, "Nmap (Network Mapper): a free open source utility for network exploration or security auditing." 2006, `http://insecure.org/nmap`.

[26] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order markov models," *Journal of Artificial Intelligence*, vol. 22, pp. 385–421, 2004.

[27] R. Stapleton-Gray and S. Gorton, "Rendering the elephant: Characterizing sensitive networks for an uncleared audience," in *Proceedings of the 7th IEEE Information Assurance Workshop*, June 2006.