AFRL-IF-RS-TR-2002-228
Final Technical Report
September 2002

ADAPTIVE, MODEL-BASED MONITORING AND THREAT DETECTION

SRI International

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. H503, J371

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

AFRL-IF-RS-TR-2002-228 has been reviewed and is approved for publication.

APPROVED: Glen E. Bahr

GLEN E. BAHR
Project Engineer

FOR THE DIRECTOR: [illegible]

WARREN H. DEBANY, Technical Advisor
Information Grid Division
Information Directorate

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 074-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503			
1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** SEPTEMBER 2002	**3. REPORT TYPE AND DATES COVERED** Final Jun 99 – Jan 02	
4. TITLE AND SUBTITLE ADAPTIVE, MODEL-BASED MONITORING AND THREAT DETECTION			**5. FUNDING NUMBERS** C - F30602-99-C-0149 PE - 62301E PR - H503 TA - 34 WU - 01
6. AUTHOR(S) Alfonso Valdes and Keith Skinner			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) SRI International Computer Science Laboratory 333 Ravenswood Avenue Menlo Park California 94025-3493			**8. PERFORMING ORGANIZATION REPORT NUMBER** N/A
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency 3701 North Fairfax Drive Arlington Virginia 22203-1714; AFRL/IFGB 525 Brooks Road Rome New York 13441-4505			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER** AFRL-IF-RS-TR-2002-228
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Glen E. Bahr/IFGB/(315) 330-3515/ Glen.Bahr@rl.af.mil			
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.			**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(Maximum 200 Words)***

We explore the suitability of model-based probabilistic techniques, such as Bayes networks, to the field of intrusion detection and alert report correlation. We describe a network intrusion detection system (IDS) using Bayes inference, wherein the knowledge base is encoded not as rules but as conditional probability relations between observables and hypotheses of normal and malicious usage. The same high-performance Bayes inference library was employed in a component of the Mission-Based Correlation effort, using an initial knowledge base that adaptively learns the security administrator's preference for alert priority and rank. Another major effort demonstrated probabilistic techniques in heterogeneous sensor correlation. We provide results for simulated attack data, live traffic, and the CyberPanel Grand Challenge Problem. Our results establish that model-based probabilistic techniques are an important complementary capability to signature-based methods in detection and correlation.

14. SUBJECT TERMS Intrusion Detection, Intrusion Correlation, Computer Network Defense, Large-Scale Attacks, Model-Based, Bayes Networks, TCP/IP, EMERALD			**15. NUMBER OF PAGES** 38 **16. PRICE CODE**
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	**18. SECURITY CLASSIFICATION OF THIS PAGE** UNCLASSIFIED	**19. SECURITY CLASSIFICATION OF ABSTRACT** UNCLASSIFIED	**20. LIMITATION OF ABSTRACT** UL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Adaptive, Model-Based Monitoring and Threat Detection

Alfonso Valdes

Keith Skinner

SRI International

333 Ravenswood Ave.

Menlo Park, CA 94070

Summary

With the expanded use of networked computers and proliferation of high-bandwidth connections, there has been an unfortunate increase in computer and network abuse. Detecting and correlating incidents of abuse is an essential aspect of information assurance. To date, intrusion detection systems have relied on matching "signatures" of known attacks to those contained in a knowledge base. This approach, while effective, misses variants of attacks as well as new attacks that are not in the knowledge base. Conversely, anomaly detection approaches have the potential for recognizing novel attacks, but in practice have been hampered by low sensitivity, lack of specificity, and unacceptable false alarm rates. This research effort explored probabilistic approaches such as Bayes systems, which encode their knowledge base not as specific signatures, but as conditional probability relations. Rather than relying on rules for metrics related to transaction control protocol (TCP) connections, the system adaptively learns these and also discovers hosts and services on the monitored network. The result is an IDS that detects many novel attacks, and aided by its adaptive capability achieves acceptable sensitivity and false alarm rates.

The number of IDS alerts in typical systems can overwhelm network security officers, raising the need for effective prioritization and correlation of alert messages. We address this need in two ways. First, in cooperative research with the SRI Mission-Based Correlation effort (supported by the same DARPA program), we reused the Bayes inference library from the TCP detector to implement an inference engine for alert ranking and priority. This capability allows the security administrator to specify preferences in a configuration file, and includes an adaptive capability that dynamically adjusts the internal knowledge base guided by administrator decisions. Additionally, we explored probabilistic techniques for alert correlation. The approach is a mix of Bayes techniques with concepts from sensor fusion. This correlation component shares the

ranking and prioritization capability of the Mission-Based Correlator. The probabilistic correlation system has proven effective with alerts that report incomplete content, or with reports from heterogeneous sensors that are at variance with each other. The system demonstrates the ability to thread an attack from probe to internal exploit as well as recognize as the same incident reports from multiple heterogeneous sensors. These capabilities have been demonstrated in our live network, in a pilot deployment at a government agency, and with the Cyberpanel Grand Challenge Problem (GCP).

Adaptive, Model-Based Monitoring and Threat Detection

1. Introduction and Overview

The field of intrusion detection has considered a variety of approaches based on anomaly detection and signature or rule-based systems. A subclass of anomaly detection systems attempts to learn behavior in a probabilistic sense. Anomaly detection systems are attractive in their ability to detect novel unusual activity, while signature systems lack this generalization potential. However, anomaly detection has enjoyed only modest success in practice, due to lack of specificity and unacceptable false alarm rates. Moreover, many important attacks do not manifest as anomalies in the features these systems observe. Signature systems have enjoyed greater success, and all leading commercial and most research systems are in this class. These systems incorporate a knowledge base that can be as simple as matching suspicious patterns in packet traffic or as sophisticated as "stateful" systems such as EMERALD [Por97].

Our objective when undertaking this research was to explore a middle ground, that is, the class of systems that incorporate a knowledge base in a probabilistic sense, giving the system some generalization potential but greater sensitivity and specificity. Bayes networks represent knowledge as conditional probability relations between observable features and hypotheses of use and misuse. Moreover, conditional probabilities are maintained as internal tables that can adaptively learn in response to new observations. The first component we implemented was a Bayes sensor for attacks visible in TCP header traffic. This proved to be effective with the Lincoln Laboratory 1999 evaluation data set [Lip00], and was further validated through extensive experimentation with live traffic. Today, this component runs live in our environment as well as at the National Security Agency (NSA).

One innovative feature of this system is that it is actually a coupled sensor, with one subcomponent that adaptively learns hosts and services on the monitored network, and another that uses this information to adjust its state dynamically. The result is improved sensitivity with a reduced false alarm rate, particularly for false alarms that are side effects of an attack or a nonmalicious failure (what we term "collateral damage"). This host availability monitor is useful in its own right in discovering new and possibly unauthorized services as well as notifying the administrator of nonmalicious failures.

Besides sensor coupling, which can be considered a form of sensor state correlation, we have explored three aspects of alert correlation. In alert threading (within-sensor correlation), the sensor maintains a concept of session, and issues alerts for suspicious sessions that consolidate many low-level events. For many attacks such as address sweeps the reduction in the number of alert messages can be two orders of magnitude. Threading is a concept common to most EMERALD sensors, but is absent from many other systems. In incident correlation, multiple reports from different and possibly heterogeneous sensors are recognized as describing the same incident. Scenario correlation chains together multiple attack steps (each a thread or incident) to reassemble more complex attacks. We have implemented a correlation engine based on probabilistic inference to accomplish these goals. The system adapts concepts from heterogeneous sensor fusion and a transition model for multistep attacks. The probabilistic approach is robust against incomplete or conflicting information from multiple sensors, which will represent the state of affairs as standards such as IDMEF [Cu01] are adopted in varying degrees. This system shares with the EMERALD Mission-Based Correlation a subsystem to prioritize and rank alerts. This subsystem in turn is based on the same Bayes inference code that underlies the Bayes TCP sensor, and adaptively learns the security administrator's preferences for alert ranking and prioritization.

The remainder of this report is organized as follows. By way of background, we provide relevant material describing Bayesian inference, including the representation of a knowledge base as conditional probability relations. We then describe the components developed in our work, namely, the Bayes TCP sensor and the probabilistic correlation module. In the description of the latter, we give an overview of the prioritization and ranking module shared with the Mission-Based Correlator. We give results of experimentation and use of these components in attack simulations as well as live traffic analysis. We then give conclusions and suggestions for further work.

2. Background

Intrusion detection to date has considered system audit trails [Val94, Lin00] and monitored network traffic [SNORT, ISS, Por97]. Most of these systems use rule-based inference ranging from simple pattern matching [SNORT, ISS] to systems that maintain some notion of session state [Por97, Lin00]. A minority employ some form of anomaly detection, including variants based on nonparametric statistics [Val94, Ski98], sequence analysis [For96], and data mining [Lee00]. To date, the rule-based systems have dominated the field, comprising all major commercial intrusion detection systems (IDS) and most research efforts. Moreover, critics of anomaly detection correctly point out that intrusions are not necessarily anomalous, and anomalies are not necessarily intrusive [McH01]. This observation raises an objection to anomaly detection in principle; additionally, in practice these systems have not shown acceptable sensitivity and sufficiently low false alert rates to gain wide acceptance. On the other hand, critics of rule-based systems point out that such systems may not be capable of detecting novel attacks. Our research explores systems where models of malicious use are not expressed as specific signatures, which are bypassable by varying the attack slightly, but are encoded in a probabilistic sense. Bayes networks [Pearl88] are particularly suitable to this representation, relating hypotheses to observable evidence by means of conditional probability relations. Our system adapts as its view changes (mathematically, by changing prior belief among competing hypotheses or modifying conditional probability relations appropriately). Our goal was to develop a system with specificity, sensitivity, and false alarm rate comparable to the better rule-based systems, but retaining some of the potential to detect novel attacks of anomaly detection. To this end, we developed a TCP session monitor capable of detecting attacks visible in TCP header data, adaptively learning system resources and parameters such as typical connection completion times. This system is probabilistic (inference is based on Bayes probability), adaptive (some parameters are learned by the system as it observes the monitored network), and model based (important classes of misuse are encoded as conditional probability models). It includes a capability based on more traditional anomaly detection that causes alerts based on extremely unusual patterns of TCP port use, but the core detection capability is Bayesian and model based [Val00].

We were also interested in the problem of IDS alert correlation, which was largely unexplored when our effort began, although there have been some contributions in the interim [De01, Hoa01]. As in the area of intrusion detection, most correlation approaches to date are based on rules and heuristics. As with the intrusion detection component, we explored a probabilistic approach to intrusion alert correlation as well. The Bayes paradigm of maintaining a prior belief over a number of hypotheses and updating this belief as new evidence is observed is used to model attack evolution over time. To assess whether a newly observed alert is plausibly connected to an existing set of correlated alerts, we define a number of features in the alert and employ an approach somewhat analogous to that used in multisensor data fusion [Hall92]. While concepts from traditional multisensor fusion provide useful guidance, definitions of such concepts as feature similarity are specific to the intrusion correlation domain. We were able to develop a successful prototype system capable of correlating alerts from heterogeneous sensors, even when the sensors disagreed as to specifics of the alert or the alert messages were improperly formed. As standards for alert interchange are still fairly immature, we feel that the inherent robustness of probabilistic systems endows them with an important advantage [Val01].

3. Methods, Approaches, and Procedures

Bayes TCP Sensor

Foundations

Mathematically, we have adapted the framework for belief propagation in causal trees from Pearl [Pearl88]. Knowledge is represented as nodes in a tree, where each node is considered to be in one of several discrete states. A node receives π (prior, or causal support) messages from its parent, and λ (likelihood, or diagnostic support) messages from its children as events are observed. We think of priors as propagating downward through the tree, and likelihood as propagating upward. These are discrete distributions, that is, they are positive valued and sum to unity. The prior message incorporates all information not observed at the node. The likelihood at terminal or "leaf" nodes corresponds to the directly observable evidence. A conditional probability table (CPT) links a child to a parent. Its elements are given by

$$CPT_{ij} = P(state = j | parent\_state = i)$$

As a consequence of this definition, each row of a CPT is a discrete distribution over the node states for a particular parent node state, that is,

$$CPT_{ij} \geq 0, \forall i, j,$$
$$\sum_j CPT_{ij} = 1, \forall j$$

The basic operations of message propagation in the tree are most succinctly expressed in terms of vector/matrix algebra. We will adopt the convention that prior messages are represented as row vectors. Downward propagation of the prior messages is achieved by left multiplication of the parent's prior by the CPT, that is,

$$\pi(node) = \alpha\pi(parent\_node) \bullet CPT$$

where α is a normalizing constant to ensure that the result sums to unity. Note that since CPT is not required to be square, the number of elements in $\pi(node)$ and $\pi(parent\_node)$ may be different. Since we limit ourselves to trees, there is at most one parent per node. However, there may be multiple children, so upward propagation of the likelihood messages requires a fusion step. For each node, the λ message, represented as a column vector, is propagated upward via the following matrix computation:

$$\lambda\_to\_parent(node) = CPT \bullet \lambda(node)$$

Note that $\lambda(node)$ has number of elements equal to the number of states in the node, while $\lambda\_to\_parent(node)$ has number of elements equal to the number of states in the parent node. These messages are fused at the parent via elementwise multiplication:

$$L_i(parent) = \Pi_{c \in children(parent)} \lambda\_to\_parent_i(c)$$
$$\lambda_i(parent) = L_i(parent) / \sum_j L_j(parent)$$

Here, L represents the raw elementwise product, and λ is obtained by normalizing this to unit sum. Finally, the belief over the states at a node is obtained as follows:

$$BEL_i = \beta\pi_i\lambda_i$$

where β is a normalizing constant so that *BEL* has unit sum. Figure 1 illustrates propagation in a fragment of a tree.

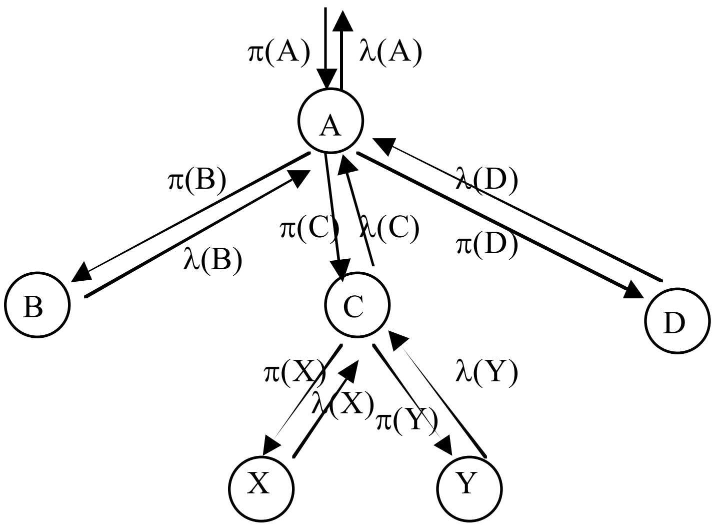

Figure 1: Message Propagation in a Tree Fragment

<u>Adaptive CPT Adjustment</u>

The system is preconfigured with CPTs relating observable features to normal and misuse hypotheses. These CPTs can adaptively evolve to adjust to specific environments. Adaptation via reinforcement proceeds as follows. We recall that the CPT relates a child node to its parent. In our representation, the rows of the CPT correspond to parent states, while the columns correspond to child states. If a single hypothesis is dominant at the root node, we adapt the corresponding row of the CPT matrix at each child slightly in the direction of the λ message at the child node for the present observation. Specifically, if hypothesis *i* "wins" at the root node, we adjust CPT as follows. First, we decay the internal effective counts via a decay function:

$$counts_i^{decay} = \gamma counts_i + (1-\gamma)$$

The decayed count is used as a "past weight" for the adjustment, and is the effective number of times this hypothesis has been recently observed. The CPT row is first converted to effective counts for each child state, and the present observation is added as an additional count distributed over the same states. Then the row elements are divided by the row sum so that the adjusted row has unit sum. This is accomplished by the following equation:

$$\mathbf{CPT}_{ij}^{adj} = \frac{counts_i \times \mathbf{CPT}_{ij} + \lambda_j}{\sum_j counts_i \times \mathbf{CPT}_{ij} + \lambda_j}$$

Finally, the internal counts are recomputed for all parent states:

$$counts_i = counts_i^{decay} + \begin{cases} \gamma, & \text{hypothesis } i \text{ is the winner} \\ 0, & \text{otherwise} \end{cases}$$

By this procedure, the effective count never decays below 1.0 (if the hypothesis is never observed) and never grows beyond $1/(1-\gamma)$ if the hypothesis is always observed. We typically choose the decay factor so that the effective count grows to between 200 and 1000 observations. Observations for frequently seen hypotheses have a smaller CPT adjustment than do observations for rare hypotheses. In addition, since only "winning" hypotheses cause a potential CPT adjustment, our system has one key advantage over other statistical ID systems. A large number of observations for a hypothesis corresponding to an attack will not be considered "normal" no matter how frequently it is observed, as its adjustment only reinforces the corresponding internal attack hypothesis model in the system.

State Transition

As a simplifying assumption, the states observed for the respective variables are considered to be independent of what was observed for these variables in past inference intervals, given the session class. In addition, given the value of the session class in the current interval, *X* is independent of any other observable variable Y. In other words, for all observable variables *X*, *Y* and inference intervals 0 to *k*, we have

$$P\left(X_k = x \middle| Sess\_class_k = s, X_{k-1} \ldots X_0, Y_{k-1} \ldots Y_0\right) = P\left(X_k = x \middle| Sess\_class_k = s\right)$$

The evolution of session class over inference intervals is modeled as a discrete time-and-state Markov process. The transition matrix is a convex combination of an identity matrix (to express state persistence) and a matrix whose rows are all equal to some prior distribution over the possible values of session class (to express the tendency of the process to decay to some prior state). In other words, for some $0 \le \gamma \le 1$, the transition matrix **M** is given by

$$\mathbf{M} = \gamma\mathbf{I} + (1-\gamma)\mathbf{P}$$

where **I** is an identity matrix and each row of **P** is given by

$$\mathbf{P}_{i,\cdot} = \mathbf{PRIOR}$$

and **PRIOR** is a prior distribution over possible values j for session class, that is,

$$\mathbf{PRIOR}_j = \text{Prior probability } (Sess\_class = j)$$

$\mathbf{M}_{ij}$ is the probability that if the process is currently in state i it will be in state j at the next event. More generally, if **POST_BEL** is our current belief state (a distribution over the possible state values, given the evidence up to and including this time interval), left multiplication with **M** redistributes our belief to obtain the prior belief before the next observation:

$$\mathbf{PRE\_BEL}_k = \mathbf{POST\_BEL}_{k-1}\mathbf{M}$$

We manipulate the parameter γ to capture, albeit imperfectly, the continuous nature of the underlying process. We typically invoke the inference function every 100 events within a session, and always when the session enters the idle state. Some sessions are less than 100 events in total, while others, particularly many denial-of-service (DOS) attacks, consist of tens of thousands of events in a very short time interval. In the latter case, even though many inference steps are invoked, we prefer to have a moderately high persistence parameter (about 0.75) because very little time has elapsed. If the parameter is 0, the belief reverts to the prior at each event.

It can be shown that, unless γ is unity, iteratively multiplying M by itself results in a matrix that approaches P, that is,

$$\lim_{n\to\infty} \mathbf{M}^n = \mathbf{P}$$

In practice, this limit is nearly reached for fairly small values of *n*. The result of this observation is attractive from the intuitive standpoint: in the absence of reinforcing evidence from subsequent events, the belief distribution tends to revert to the prior.

The inference operation at interval *k* begins by setting the Bayes π message to $\mathbf{PRE\_BEL}_k$. Then the observables over the interval are presented to the leaf nodes, and the belief state at the root node is extracted. If this is deemed sufficiently suspicious, the system generates an alert message that can be displayed at a console or forwarded to a correlation utility.

Probabilistic Correlation

Our probabilistic correlation approach considers feature overlap, feature similarity, minimum similarity, and expectation of similarity. In this context, a "feature" is the value of a field in the alert that is pertinent to alert correlation. Features include source and target network addresses and ports, the type of attack, and the attack time. We maintain a list of "meta alerts" that are possibly composed of several alerts, potentially from heterogeneous sensors. For two alerts (typically a new alert and a meta alert), we begin by identifying features they have in common (feature overlap). Such features include the source of the attack, the target (hosts and ports), the class of the attack, and time information. With each feature, we have a similarity function that returns a number between 0 and 1, with 1 corresponding to a perfect match. Similarity is a feature-specific function that considers such issues as

- How well do two lists overlap (for example, list of targeted ports)?
- Is one observed value contained in the other (for example, is the target port of a DOS attack one of the ports that was the target of a recent probe)?
- If two source addresses are different, are they likely to be from the same subnet?

For attack class similarity, we maintain a matrix of similarity between attack classes, with values of unity along the diagonal and off-diagonal values that heuristically express similarity between the corresponding attack classes. We prefer to consider attack classes rather than attack signatures, which are much more specific and numerous but may be erroneously or incompletely reported. For example, in our demonstration environment, we run a variant of mscan that probes certain sensitive ports, that is, it is of the attack class "portsweep". Our host sensors have a specific signature for this attack and call it "mscan". The Bayes sensor trades specificity for generalization capability and has no "mscan" model, but successfully detects this attack as a "portsweep". These reports are considered similar ($S = 1$) with respect to attack class.

Not all sensors produce all possible identifying features. For example, a host sensor provides process identifier, while a network sensor does not. Features not common to both alerts are not considered for the overall similarity match.

The meta alert itself supports the threading concept, so we can visualize composing meta alerts from meta alerts.

Similarity Expectation and Minimum Similarity

An important innovation we introduce is expectation of similarity. As with similarity, this is also between 0 and 1, and expresses our prior expectations that the feature should match if the two alerts are related, considering the specifics of each. For example, two probes from the same target might scan the same set of ports on different parts of our subnet (so expectation of matching target IP address is low). Also, some attacks such as SYN FLOOD spoof the source address, so we would allow a match with an earlier probe of the same target even if the source does not match (expectation of match for source IP is low).

We now give some examples of how expectation of similarity depends on the situation, that is, the features in the meta alert and the new alert.

If an alert from a sensor has a thread identifier that matches the list of sensor/thread identifiers for some meta alert, the alert is considered a match and fusion is done immediately. In other words, the individual sensor's determination that an alert is an update of or otherwise related to one of its own alerts overrides other considerations of alert similarity.

If the meta alert has received reports from host sensors on different hosts, we do not expect the target host feature to match. If at least one report from a network sensor has contributed to the meta alert and a host sensor alert is received, the expectation of similarity is that the target address of the latter is contained in the target list of the former.

In determining whether an exploit can be plausibly considered the next stage of an attack for which a probe was observed, we expect the target of the exploit (the features host and port) to be contained in the target host and port list of the meta alert.

Some sensors, particularly those that maintain a degree of state, report start and end times for an attack, while others can only timestamp a given alert. The former deal with time intervals, while the latter do not. Similarity in time comprehends overlap of the time intervals in the alerts considered for correlation, as well as the notion of precedence. We do not penalize time similarity too far from unity if the time difference is plausibly due to clock drift.

Deciding whether the attacker is similar is somewhat more involved. In the case of an exact match of originating IP address, similarity is perfect. We assign high similarity if the subnet appears to match. In this way, a meta alert may potentially contain a list of attacker addresses. At this point, we consider similarity based on containment. In addition, if an attacker compromises a host within our network (as inferred by a successful outcome for an attack of the root compromise class), that host is added to the list of attacker hosts for the meta alert in question. Finally, for attack classes where the attacker's address is likely to be spoofed (for example, the Neptune attack), similarity expectation with respect to attacker address is assigned a low value.

Our correlation component implements not just expectation of similarity (which effectively acts as a weight vector on the features used for similarity matching) but also enforces situation-specific minimum similarity. Certain features can be required to match exactly (minimum similarity for these is unity) or approximately (minimum similarity is less than unity, but strictly positive) for an alert to be considered as a candidate for fusion with another. Minimum expectation thus expresses necessary but not sufficient conditions for correlation.

The overall similarity between two alerts is zero if any overlapping feature matches at a value less than the minimum similarity for the feature (features for which no minimum similarity is specified are treated as having a minimum similarity of 0). Otherwise, overall similarity is the weighted average of the similarities of the overlapping features, using the respective expectations of similarity as weights.

Correlation Modes

By appropriate settings of similarity expectation and minimum similarity, the correlation component achieves the following hierarchy of correlation. The system is composable in that we can deploy multiple instances to obtain correlation at different stages in the hierarchy. For example, we can infer threads (within sensor correlation) and then correlate threaded alerts from heterogeneous sensors into security incidents.

Synthetic Threads: For sensors that do not employ the thread concept, the correlation synthesizes threads by enforcing high minimum expectation similarity on the sensor itself (the thread must come from a single sensor) and the attack class, as well as source and target (IP and ports). We have wrapped the alert messages from a leading commercial sensor and observed that this facility reliably reconstructs threads.

In particular, by placing an aggregator component topologically close to an IDS, the pair is made robust against attacks that cause the IDS itself to flood, as described in a recent NIPC advisory [NIPC01].

Security Incidents: By suppressing minimum expectation of similarity on the sensor identifier, and relaxing expectation of similarity for this feature, we can fuse reports of the same incident from several heterogeneous sensors into a single incident report. In this case, we enforce a moderately high expectation of similarity on the attack class. This is not unity because different sensors may report a different attack class for the same attack. We construct a table of distances between attack classes that expresses which ones are acceptably close. For security incident correlation, we enforce minimum expectations on the source and target of the attack. Using this technique, we have been able to fuse alert reports from commercial and EMERALD sensors into security incident reports.

Correlated Attack Reports: By relaxing the minimum expectation of similarity on the attack class, we are able to reconstruct various steps in a multistage attack. Each stage in an attack may itself be a correlated security incident as described above. In this fashion, it is possible to recognize a staged attack composed of, for example, a probe followed by an exploit to gain access to an internal machine, and then using that machine to launch an attack against a more critical asset.

Feature Fusion

When the system decides to fuse two alerts, based on aggregate similarity across common features, the fused feature set is a superset of the features of the two alerts. Feature values in fused alerts are typically lists, so alert fusion involves list merging. For example, suppose a probe of certain ports on some range of the protected network matches in terms of the port list with an existing probe that originated from the same attacker subnet, but the target hosts in the prior alert were to a different range of our network. The attacker address list has the new attacker address appended, and the lists of target hosts are merged. The port list matches and is thus unchanged.

Two important features are the sensor and thread identifiers of all the component alerts, so that the operator is always able to examine in detail the alerts that contribute to the meta alert report.

One additional feature is the priority of the meta alert, supported by our template and provided by EMERALD sensors. We are developing a component that estimates criticality based on the assets affected, the type of attack, the likelihood of attack success, and an administrative preference. The aggregator maintains the high-water mark for this field. We are investigating approaches whereby the contributing threads are permitted to update their priority downward, computing meta alert priority as the maximum across thread priorities at any given time. This approach would permit downward revision of the meta alert priority.

The features presently considered in the probabilistic correlator component include sensor identification (identifier, location, name), alert thread, incident class, source and target IP lists, target TCP/UDP port lists, source user id, target user id, and time. Computations are only over features that overlap in the alert to be merged and the candidate meta alert into which it is to be merged. Incident signature is used as well, but with a low expectation of similarity as these vary widely across heterogeneous sensors.

If present, a thread identifier from the reporting sensor overrides other match criteria. A new alert that matches the sensor and thread of an existing meta alert is considered an update of the earlier alert.

The correlator first tries to infer a thread by looking for an exact match in sensor identification and incident class and signature. Note that alerts that are inferred to be from the same thread may be separated in time. The system attempts to infer threads even in incident and scenario operational modes.

Next the system checks that all overlapping features match at least at their minimum similarity value. Setting minimum expectation for some features to unity (not normally recommended) causes the system to behave like a heuristic system that requires exact matches on these features. Given that this criterion passes, we compute the overall similarity between the two alerts as follows:

$$SIM(X,Y) = \frac{\sum_j E_j SIM(X_j, Y_j)}{\sum_j E_j}$$

$X =$ Candidate meta alert for matching

$Y =$ New alert

$j =$ Index over the alert features

$E_j =$ Expectation of similarity for feature j

$X_j, Y_j =$ Values for feature j in alerts X and Y, respectively (may be list valued)

Incident class similarity is based on a notion of proximity, which at present is the result of our judgment. The proximity of class A to B reflects how reasonably an attack currently of incident class A may progress to class B. Note that this is not symmetric; we more strongly expect an exploit to follow a probe than the other way around. The incident classes shown in Table 1 are from the EMERALD 602 message format. Note that some "default" classes such as "invalid" and "action logged" are reasonably proximal to most other classes. This occurs because the IETF standard does not require a common ontology, and reports from heterogeneous sensors for the same incident may not reliably represent this field. As such, we do not want to reject potential matches based on this field alone.

For operational modes other than thread level aggregation, we do not recommend a high minimum similarity value for this field.

	INVALID	PRIVILEGE_VIOLATION	USER_SUBVERSION	DENIAL_OF_SERVICE	PROBE	ACCESS_VIOLATION	INTEGRITY VIOLATION	SYSTEM_ENV_CORRUPTION	USER_ENV_CORRUPTION	ASSET_DISTRESS	SUSPICIOUS_USAGE	CONNECTION_VIOLATION	BINARY_SUBVERSION	ACTION_LOGGED
INVALID	1	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.6
PRIVILEGE_VIOLATION	0.3	1	0.6	0.3	0.6	0.6	0.6	0.6	0.4	0.3	0.4	0.1	0.5	0.6
USER_SUBVERSION	0.3	0.6	1	0.3	0.6	0.5	0.5	0.4	0.6	0.3	0.4	0.1	0.5	0.6
DENIAL_OF_SERVICE	0.3	0.3	0.3	1	0.6	0.3	0.3	0.4	0.3	0.5	0.4	0.1	0.5	0.6
PROBE	0.3	0.2	0.2	0.3	1	0.7	0.3	0.3	0.3	0.3	0.4	0.8	0.3	0.6
ACCESS_VIOLATION	0.3	0.6	0.3	0.5	0.6	1	0.6	0.6	0.3	0.3	0.4	0.1	0.5	0.6
INTEGRITY VIOLATION	0.3	0.5	0.3	0.5	0.6	0.8	1	0.6	0.5	0.3	0.4	0.1	0.5	0.6
SYSTEM_ENV_CORRUPTION	0.3	0.5	0.3	0.5	0.6	0.6	0.6	1	0.6	0.3	0.4	0.1	0.5	0.6
USER_ENV_CORRUPTION	0.3	0.5	0.5	0.3	0.6	0.6	0.6	0.6	1	0.3	0.4	0.1	0.5	0.6
ASSET_DISTRESS	0.3	0.3	0.3	0.6	0.3	0.3	0.3	0.3	0.3	1	0.4	0.4	0.3	0.6
SUSPICIOUS_USAGE	0.3	0.3	0.5	0.3	0.5	0.6	0.5	0.6	0.5	0.3	1	0.1	0.3	0.6
CONNECTION_VIOLATION	0.3	0.1	0.1	0.3	0.8	0.3	0.3	0.3	0.3	0.5	0.4	1	0.3	0.6
BINARY_SUBVERSION	0.3	0.3	0.3	0.3	0.3	0.6	0.6	0.6	0.5	0.3	0.4	0.1	1	0.6
ACTION_LOGGED	0.3	0.3	0.3	0.3	0.6	0.5	0.3	0.3	0.3	0.3	0.4	0.3	0.3	1

Table 1: Incident Class Similarity Matrix

For two alerts that are extremely close in time, it is possible that the alerts may not be in time order. In this case, incident class similarity is the greater of SIM(X, Y) and SIM Y, X). Mathematically, the similarity computation for incident class can comprehend a discrete call (the alert is from one of the above classes) or a call that is a probability distribution over the above classes (as might result from a meta alert in which the contributing sensors do not agree on the class).

Most other features are potentially list valued. For lists, the notion of similarity generally expresses the fraction of the smaller list that is contained in the larger. For source IP addresses, similarity also attempts to express the notion that the addresses in question may come from the same subnet.

Time similarity is a step function that drops to 0.5 after one hour. A close match in time is expected only when the system operates in "incident" mode. Thread and scenario aggregation may be over time intervals of days.

Correlated Alert Prioritization

In addition to rationally aggregating related alerts, a correlation system should also assign priorities to the correlated alert. Given a network topology and a preference for ranking different classes of attacks as more or less critical, a security expert can accomplish this priority assignment, but the process is manual and time consuming. We developed a Bayes system (reusing the inference library from the Bayes TCP sensor previously described) to duplicate the priorities that a security expert would assign to a given set of alerts. This system enables the following functions:

- Ability to weight the priority ranking along several attribute groupings, such as attack type or criticality of assets affected.
- Compact representation of the influence of the value of an attribute on the priority assigned.
- Incorporation of the administrator's preference profile as to the relative importance of observed values (such as attack type).
- Ranking influenced only by those attributes specified on a given alert — in general, a given alert may not observe all possible attributes.
- Ability to update the ranking based on observation of a new attribute.
- Extensibility of the model to comprehend attributes that may be defined in the future, with minimal perturbation to the rest of the model.

Computationally, our approach is to design a Bayes classifier whose output is a ranking value and whose observable evidence consists of the attribute values. The influence of an attribute on the output is expressed in terms of conditional probability relations.

Bayes approaches and probabilistic formalisms in general represent a minority of methodologies employed to date by intrusion detection systems as well as evolving systems for correlating and prioritizing alerts from such systems. Theoretically, a probabilistic system needs to specify the entire joint probability distribution of observable attributes and corresponding priority ranking. This is extremely difficult because of the "curse of dimensionality." Instead, the Bayes approach is to assume that dependencies between attributes are local, so a much more compact representation of the system's knowledge base (local conditional probability relations) is possible. The compactness of knowledge representation and the adaptive potential make this approach attractive relative to signature systems.

4. Alternative Approaches and Evaluation

Another form of adaptation is the potential ability to add a state, that is, a hypothesis representing a new mode of usage. Naïve Bayes models such as the one described above work well in practice as classifiers, and are typically trained with observations for which the true class is known. Dynamic hypothesis generation as described here takes on a more difficult problem, namely, the situation where the data cases are unlabeled and even the underlying number of hypothesis states is unknown. In this situation, it is legitimate to ask if a system can self-organize to a number of hypotheses that adequately separate the important data classes. In this respect, the ability to separate attack classes A and B from each other is less important than the ability to separate both A and B from the set of nonattack classes.

To build this capability, we need to enable the system to add hypotheses at the root node (the reader will recall that the root node state value is not directly observable). As a configuration option, the system will create a "dummy state" at the root node (or more generally, at any node that is not directly observable), with an effective count of 1. If this node has children, a new CPT row is added at each child. We use a uniform distribution over the child state (each element has value $1/nstate_{child}$) for this CPT at present.

Adding a state then proceeds as follows. The inference mechanism is applied to an observation, and a posterior belief is obtained for the dummy state as if it were a normal state. If this state "wins", it is promoted to the valid state class and the CPT rows for all children are modified via the CPT adjustment procedure described above. Note that since the effective count of the dummy state is 1, the adjustment makes the CPT rows look 50% like the observation. Then a new dummy state is added, allowing the system to grow to the number of root node states that adequately describe the data. This dummy state is not to be confused with the OTHER ATTACK hypothesis, for which there is an initial model of nonspecific anomalous behavior (e.g., moderate error intensity).

There are two ways to exploit the hypothesis generation capability. In the first, we initialize the system with the normal and attack hypotheses described above, using CPTs derived from our own domain expertise. We observe that the system does adjust the CPTs somewhat, but does not choose to add more hypotheses when running in this fashion. From this, we tentatively conclude that no more than 12 hypotheses are needed to classify these data.

Our next experiment examined the other extreme. We initialized the system with a single valid hypothesis and a dummy hypothesis at the root node. We then presented a week of normal (attack-free) data, and the system generated two valid states. As these states were generated, the CPTs were adjusted according to the procedure previously outlined. We then arbitrarily decided that any new states learned would be reported as potential attacks, and presented data known to contain attacks. The system added two new states, which captured the attacks seen previously by the 11-state expert-specified model. Therefore, with the capabilities of adaptation via reinforcement as well as state space expansion described above, it is in fact possible to start the system with essentially no initial knowledge. It then organizes to an appropriate number of hypotheses and CPT values. It is interesting that this system does nearly as well at separating the important classes (here, attack versus nonattack) as the expert-specified model with only four root node hypothesis states. Normal data is adequately represented by two states, and the variety of attack data by two abnormal states. While this does tend to separate important normal and attack classes into separate hypotheses, explaining the result is more difficult. Nonetheless, this minimal knowledge approach does remarkably well, and is a very favorable indicator of the generalization potential of our methodology.

The learning procedures described above have proven useful in our experimentation, guiding us both in refinement of existing hypotheses as well as developing new hypotheses for both normal and attack modalities. However, we have observed better operation if the adaptive capability is disabled, for several reasons. First, attacks and alert-worthy events are a very small fraction of total traffic in a real-world setting, so that learning an attack modality that may be seen only once is problematic. Second, we found that the normal hypotheses become “hardened” so as to be relatively intolerant of erroneous outcomes. The fraction of such outcomes for nonmalicious reasons is too high to be tolerable from an alert standpoint, but is too low to permit sufficient “breathing room” if adaptation is permitted indefinitely. For the present, therefore, we run the system in adaptive mode to identify unanticipated modalities and large CPT deviations from what is observed in true traffic. We then take the results of this phase and moderate it with our judgment (sanding the corners off very hardened hypotheses, so to speak) and arrive at a batch specification of the CPT. We then verify that this new encoding remains sensitive against simulated datasets (such as the Lincoln data). At present, we detect the most attacks we have ever detected in the Lincoln data, and detect alert-worthy events in our real-world data with an acceptable level of apparent false alerts.

5. Proposed Solution

Bayes TCP Sensor and Host Availability Monitor

The probabilistic methodologies presented above present an important complement to heuristic and rule-based systems for both detection and alert correlation. We explored a number of variants in the Bayes TCP sensor, such as adaptive CPT adjustment dynamic hypothesis generation. While these proved to be interesting capabilities, we must at this point consider them research features and do not make them active by default in the production version. We instead employ a system that considers simultaneous TCP sessions, maintains a Bayes hypothesis that classifies this session into one of a number of normal or misuse categories, transitions this session state over time, and evaluates the state as new evidence is observed.

We have developed eBayes as a part of the broad EMERALD system, which permits us to leverage from a substantial component infrastructure. Specifically, it is an analytical component that interfaces to the EMERALD ETCPGEN and EMONTCP components. ETCPGEN can process either live TCP traffic or TCPDUMP data in batch mode. EMONTCP extracts the TCP state for a number of generally simultaneous TCP connections. When we refer to “events”, we mean events from EMONTCP, which already represents a considerable reduction from the raw TCP data. There are two components in eBayes: the session monitor, and the host availability monitor.

The first of these components analyzes TCP sessions, which are imperfectly described as temporally contiguous bursts of traffic from a given client IP. We say “imperfectly” because it is not very important for the system to demarcate sessions exactly. The analysis is done by Bayesian inference at periodic intervals in a session, where the interval is measured in number of events (inference is always done when the system believes that the session has ended). Between inference intervals, the system state is propagated according to a Markov model.

The second component discovers what services are advertised within the domain monitored by eBayes, and then adapts to traffic intensity and connection failure rates. It continuously estimates belief in the operational state of these services, generating alerts when a service failure is apparent. As such, it can potentially detect a coordinated, distributed attack where no session appears sufficiently anomalous to the session monitor. It can also detect failures due to nonmalicious faults.

The innovation provided by eBayes is that it captures the best features of signature-based intrusion detection as well as anomaly detection (as in EMERALD eStat). Like signature engines, it can embody attack models, but has the capability to adapt as systems evolve. Like probabilistic components, it has the potential to generalize to previously unseen classes of attacks. In addition, the system includes an adaptive capability, which can "grow" quite reasonable models from a random start. However, since it has major attack classes encoded in its conditional probability tables, it can provide effective detection "out of the box".

This system detects a variety of scans and sweeps as well as flood attacks. It does not examine packet payload, but is limited to attacks that are visible in the packet headers. The session logic achieves alert threading in the sense of aggregating a small number of reports from attacks that manifest as a large number of raw events, which is typical of floods and some probes.

Probabilistic Correlation

Our probabilistic alert fusion approach considers feature overlap, feature similarity, minimum similarity, and expectation of similarity. We maintain a list of "meta alerts" that are possibly composed of several alerts, potentially from heterogeneous sensors. For two alerts (typically a new alert and a meta alert), we begin by identifying features they have in common (feature overlap). Such features include the source of the attack, the target (hosts and ports), the class of the attack, and time information. With each feature, we have a similarity function that returns a number between 0 and 1, with 1 corresponding to a perfect match.

Expectation of similarity is also a number between 0 and 1, and expresses our prior expectations that the feature should match if the two alerts are related, considering the specifics of each. We can consider expectation of similarity as a feature weighting that can vary based on the type of correlation being performed.

If an alert from a sensor has a thread identifier that matches the list of sensor/thread identifiers for some meta alert, the alert is considered a match and fusion is done immediately. In other words, the individual sensor's determination that an alert is an update of or otherwise related to one of its own alerts overrides other considerations of alert similarity.

If the meta alert has received reports from host sensors on different hosts, we do not expect the target host feature to match. If at least one report from a network sensor has contributed to the meta alert and a host sensor alert is received, the expectation of similarity is that the target address of the latter is contained in the target list of the former.

In determining whether an exploit can be plausibly considered the next stage of an attack for which a probe was observed, we expect the target of the exploit (the features host and port) to be contained in the target host and port list of the meta alert.

Some sensors, particularly those that maintain a degree of state, report start and end times for an attack, while others can timestamp only a given alert. The former deal with time intervals, while the latter do not. Similarity in time comprehends overlap of the time intervals in the alerts considered for correlation, as well as the notion of precedence. We do not penalize time similarity too far from unity if the time difference is plausibly due to clock drift.

Deciding whether the attacker is similar is somewhat more involved. In the case of an exact match of originating IP address, similarity is perfect. We assign high similarity if the subnet appears to match. In this way, a meta alert may potentially contain a list of attacker addresses. At this point, we consider similarity based on containment. In addition, if an attacker compromises a host within our network (as inferred by a successful outcome for an attack of the root compromise class), that host is added to the list of attacker hosts for the meta alert in question. Finally, for attack classes where the attacker's address is likely to be spoofed (for example, the Neptune attack), similarity expectation with respect to attacker address is assigned a low value.

Our correlation component also enforces situation-specific minimum similarity. Certain features can be required to match exactly (minimum similarity for these is unity) or approximately (minimum similarity is less than unity, but strictly positive) for an alert to be considered as a candidate for fusion with another. Minimum expectation thus expresses necessary but not sufficient conditions for correlation.

The overall similarity between two alerts is zero if any overlapping feature matches at a value less than the minimum similarity for the feature (features for which no minimum similarity is specified are treated as having a minimum similarity of 0). Otherwise, overall similarity is the weighted average of the similarities of the overlapping features, using the respective expectations of similarity as weights.

By appropriate settings of similarity expectation and minimum similarity, the correlation component achieves a hierarchy of correlation into threads, incidents, and scenarios. The system is composable in that we can deploy multiple instances to obtain correlation at different stages in the hierarchy. For example, we can infer threads (within sensor correlation) and then correlate threaded alerts from heterogeneous sensors into security incidents. The correlation component can function in thread, incident, and scenario modes, and the modes may be run concurrently.

6. RESULTS AND DISCUSSION

Bayes TCP Sensor

Lincoln Laboratory 1999 Evaluation Study

We have run our model against the TCP dump data from the 1999 Lincoln Laboratory IDEVAL data sets [Lip00]. It is highly effective against floods and nonstealthy probe attacks, and moderately effective against stealthy probe attacks.

This data simulates activity at a medium-size LAN with typical firewalls and gateways. Traffic generators simulate typical volume and variety of background traffic, both intra-LAN and across the gateway. Attack scripts of known types are executed at known times, and the traffic (a mix of normal background as well as attack) is collected by standard utilities, such as TCPDUMP.

For this prototype we examined external-to-internal traffic using the TCP/IP protocol. This means that console attacks, insider attacks, and attacks exploiting other protocols such as IDP and UDP are invisible. These are not theoretical limitations, and we intend to include the UDP protocol in the near future. However, this did limit attacks that were visible to the system. The fourth week of the data set was considered the most difficult, as it contained the most stealthy attacks. We detected three visible portsweeps and missed one that accessed three ports over four minutes with no errors. All of the portsweeps in this data set are stealthy by the standards of the Lincoln training data and the week 5 data (we detect 100% of visible, nonstealthy sweeps). A Satan attack and a TCPRESET attack are also detected as portsweeps. This particular Satan attack was run in a mode where it in fact is characteristic of a portsweep. For the TCPRESET, the portsweep hypothesis slightly edges out the OTHER hypothesis. Other detected attacks in this data include MAILBOMB and PROCESS TABLE (both 100% detected) as well as three password-guessing attacks (one detected as OTHER, two as DICTIONARY). The latter three detections demonstrate the power of the approach. They were not in the set of attacks that Lincoln thought should be detected by this sensor, so we initially considered them false alarms. Further review of the full attack list indicated that they were in fact good detections, even though at that time we had no DICTIONARY hypothesis and they were called OTHER. By elucidating characteristics of these attacks, we added the DICTIONARY hypothesis (indicative of password guessing), which now captures two of these attacks and is a close second to OTHER as a classification for the third.

Real-World Experience

The Bayes TCP component runs on our own TCP gateway, and it has proved to be stable for indefinite periods of time. The TCP event generator, EMONTCP, and Bayes inference components require about 15MB on a Free BSD platform, and never use more than a few percent of the CPU. For real-world traffic, we of course have no ground truth, but the results have nonetheless proved interesting to us in the sense of scientific experimentation, as well as being of practical interest to our system administrators.

Our initial observation was that, not surprisingly, real-world data contains many failure modes not seen in a set such as the IDEVAL data described above. For example, we regularly observe a pattern of http sessions of moderate or long duration in which a significant number of connections terminate abnormally, but on such a time scale and in such modes that we are fairly certain they are not malicious. To capture these sessions, we decided to add the HTTP_F hypothesis (for failed http). This reduced the alert volume to a manageable 15 or so per day. A representative two-week period comprised about 470,000 connection events, grouped by the session model into about 60,000 sessions of which 222 produced alerts. It is important to point out that many of these are almost certainly attacks, consisting of IP and probe sweeps and some attempted denials of service. Some of the false alert mechanisms are understood and we are actively working to improve system response to these without being too specific (for example, ignoring alerts involving port 113 requests, which are screened in our environment but will be seen from normal mail clients).

Probabilistic Correlator

Live Traffic

The following is an example of alert correlation over time, in this case correlating alerts that are components of a stealthy port sweep. The following example is one of the contributing alerts. In the interest of space, we do not include all the content that is in the alert and meta alert templates, but limit ourselves to the fields needed to illustrate the result.

```
Thread ID 69156 Class= portsweep   BEL (class) =  0.994 BEL(attack)=  1.000
2001-06-15 17:34:35 from xx.yyy.148.33 ports 1064 to 1066 duration=   0.000
dest IP aaa.bbb.30.117
3 dest ports: 12345{2} 27374{3} 139
```

This is a probe for three vulnerable ports on a single IP address in the protected network, and is detected by the Bayes TCP sensor. The example above is just a single step in a probe that apparently transpired over several days, and resulted in the following correlated meta alert.

```
Meta Alert Thread 248
Source IPs source_IParray: xx.yyy.148.33 xx.yyy.148.47

Target IPs target_IParray: aaa.bbb.30.117 aaa.bbb.6.232 aaa.bbb.8.31
aaa.bbb.1.166 aaa.bbb.7.118 aaa.bbb.28.83 aaa.bbb.19.121 aaa.bbb.21.130
aaa.bbb.6.194 aaa.bbb.1.114 aaa.bbb.16.150

From  2001-06-15 17:34:35 to  2001-06-21 09:19:57
correlated_alert_priority -1

Ports target_TCP_portarray: 12345{4} 27374{4} 139{3}

Number of threads 10 Threads :69156 71090 76696 84793 86412 87214 119525
124933 125331 126201
Fused: PORT_SCAN
```

We note that we have correlated events from two source addresses that were judged to be sufficiently similar. The attack is quite stealthy, consisting of a small number of attempted connections to single target hosts over a period of days. The list of thread identifiers permits the administrator to examine any of the stages in the attack. In this case, each attack stage is considered a portsweep; if the stages consisted of different attack classes, these would be listed under “Attack steps”. Over the three-week time period containing this attack, the IDS sensor processed more than 200,000 sessions and generated 4439 alerts. The probabilistic correlation system produced 604 meta alerts.

In a more recent live traffic analysis experiment, we considered alerts from several EMERALD sensors as well as SNORT, operating in August 2001 during the height of the Code Red and Code Red II attacks. The probabilistic correlation engine considers attack class as one of the features in its similarity matching algorithms. The mapping between attack classes and attack signatures is implemented in the EMERALD incident handling knowledge base (IHKB), which is shared by all EMERALD sensor and correlation components. Currently, all signatures are mapped into the 14 IHKB classes listed in Table 2.

ACCESS VIOLATION	DENIAL OF SERVICE	SUSPICIOUS USAGE
ACTION LOGGED	INTEGRITY VIOLATION	SYSTEM ENVIRONMENT CORRUPTION
ASSET DISTRESS	INVALID	USER ENVIRONMENT CORRUPTION
BINARY SUBVERSION	PRIVILEGE VIOLATION	USER SUBVERSION
CONNECTION VIOLATION	PROBE	

Table 2: EMERALD IHKB Incident Classes

Due to time constraints, we were not able to populate the mapping of SNORT alerts to EMERALD incident classes, so SNORT alerts are assigned to a fallback “ACTION LOGGED” class. An advantage of probabilistic techniques is that this approach produces slightly lower fidelity results, but the technique is sufficiently robust to tolerate this as a minor deficiency.

Because of the overall architecture of EMERALD, we are able to deploy a correlation capability at one or more points in a monitoring network, and can in fact correlate correlated alerts. We chose to separately correlate the SNORT alerts, the EMERALD alerts, and the entire set. The first function of correlation, as presented in the introduction, is to reduce the raw number of alert reports that a security administrator must examine. Table 3 reflects totals for a one-day collection period in our laboratory, starting at 10 a.m. PDT, August 6-7, 2001.

Sensor	Raw Alerts	Correlated Alerts
Snort	4816	487
EMERALD	1586	523
Composite	6402	869

Table 3: Heterogeneous Sensor Correlation Live Traffic Results

As described above, it is possible that a raw alert will fail to correlate with any other alerts. In this case, the corresponding correlated alert will consist solely of the contents of the single contributing raw alert. Therefore, the set of correlated alerts contains information for all of the raw alerts. We observe that correlation achieves about a 10 to 1 reduction in SNORT alerts, and about 3 to 1 for EMERALD alerts. This occurs because the EMERALD sensors attempt to thread alerts, as we have previously discussed.

Cyberpanel Grand Challenge Problem

The Cyberpanel Grand Challenge Problem (GCP) was formulated to facilitate experimentation with alert correlation systems. The goal was to present to correlation systems a set of alerts that were realistic in the sense of the volume and nature of alerts, containing many nuisance attacks and one critical attack scenario. The objective of the developer of a correlation methodology was to correlate the nuisance and critical alerts, thereby reducing total alert volume to a more manageable level, and to identify alerts related to the critical attack as representing something more serious than the background nuisance traffic. It is also crucial that the alerts from the critical attack not spuriously correlate with alerts from the nuisance attacks. To prioritize alerts, we activated the alert prioritization functionality, which is based on the same Bayes inference library as the TCP sensor and is shared by the EMERALD MCorrelator as well.

Table 4 summarizes alert reduction results. The “truth” files contain alerts representing the critical attacks, while the “all” files contain the same critical attack alerts and a large number of nuisance attack alerts.

	Attack 1	Attack 2
Truth		
Total	117	8
Correlated	13	3
All		
Total	7216	7634
Correlated	474	475

Table 4: Grand Challenge Problem Correlation Results

As desired, the alerts representing the critical attacks did not correlate with the other alerts. Moreover, the correlated alerts representing critical attack scenarios were assigned priorities of 240 (on a 0 to 255 scale) while other alerts were scored below 127.

7. Conclusions and Suggestions for Future Work

We have developed components for intrusion detection and intrusion report correlation that use probabilistic techniques rather than the more common signature and heuristic approaches used in the field. We believe that they are not a replacement for the latter approaches, but do provide important complementary capabilities in the areas of generalization potential, adaptability to changing conditions, and robustness against improperly formed or conflicting messages.

The intrusion detection components consist of a TCP session monitor and a closely coupled host availability monitor, both based on Bayes inference. The former detects a variety of attacks visible in TCP packet headers, while the latter discovers new network hosts and services and detects failures (malicious or not). By coupling these sensors, the sensitivity and false alarm rate of the overall system are greatly improved.

The probabilistic correlation component adapts concepts from multisensor data fusion and introduces innovative similarity functions suitable to the IDS alert correlation domain. It includes a Bayes subsystem that reproduces the priority assignment that an expert security administrator would give to a set of alerts. The probabilistic approach is robust in the heterogeneous sensor environment, where sensors may not agree about the particulars of a given attack, and some sensors may implement alert interchange standards incompletely.

We have extensive experimental and live experience with both systems. Also, these systems operate against live traffic at an Internet gateway for the NSA. Their experience and our own has enabled continuous refinement of the components, so that they now achieve impressive results with high stability.

In terms of future work, we would like to explore the scenario of cross-domain correlation. This is motivated somewhat by the grand challenge problem, and addresses the issue of a simultaneous attack against multiple autonomous but cooperating domains. The scenario is appropriate to a distributed command mission, as well as potentially to civilian infrastructure and homeland defense.

We would also like to explore synergies between our correlation work and the Correlated Attack Modeling (CAM) effort. Specifically, we would represent CAM models as a special class of meta alert that is essentially a template, with appropriate wildcards for feature matching. These would form a special set of "seed" alerts in the meta alert list. A set of alerts that match the seed alert then initiate a correlated alert of the corresponding correlated attack type.

We are actively pursuing opportunities to transition this technology into the OPX Analyst Work Bench (AWB), as well as to the U. S. Army and the Federal Aviation Administration.

Acknowledgements
We are grateful to our sponsors, DARPA and Air Force/Rome Laboratory, for their support and guidance in this effort, as well as for providing technology transition opportunities as the work matured. We would like to thank our colleagues at other sites supporting the Cyberpanel effort, especially Josh Haines at Lincoln Laboratory, for their useful input. We would also like to thank the EMERALD team. Phil Porras and Martin Fong contributed much of the conceptual development of the correlated alert prioritization module. Discussions with Ulf Lindqvist and Steven Cheung have helped us solidify our approach to probabilistic correlated attack modeling, discussed above as interesting future work. Other EMERALD team members are Magnus Almgren, John Khouri, and Sandy Smith.

REFERENCES

1. [Cu01] Curry, D. and Debar, H. "Intrusion Detection Message Exchange Format", http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-03.txt
2. [De01] Debar, H. and Wespi, A. "Aggregation and Correlation of Intrusion-Detection Alerts", *Recent Advances in Intrusion Detection (RAID) 2001*, Davis, CA, 10-12 October 2001. Springer-Verlag Lecture Notes in Computer Science, October 2001.
3. [For96] Forrest, S. Hofmeyr, S., Somayaji, A., and Longstaff, T. "A Sense of Self for Unix Processes*", Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy*, pp. 120-128, IEEE Computer Society Press, May 1996.
4. [Hall92] Hall, D. "Mathematical Techniques in Multisensor Data Fusion", Artech House, Boston, MA. 1992.
5. [Hoa01] Hoagland, J., McAlemy, J., and Stanniford, S. "Practical Automated Detection of Stealthy Portscans", http://www.securityfocus.com/library/3019
6. [Lee00] Lee, W., Nimbalkar, R., Yee, K., Patil, S., Desai, P., Tran, T., and Stolfo, S. "A Data Mining and CIDF Based Approach for Detecting Novel and Distributed Intrusions", Proceedings of *Recent Advances in Intrusion Detection* (*RAID),* Toulouse, France, 2-4 October, 2000, Springer-Verlag Lecture Notes in Computer Science, October, 2000, pp. 49-65.
7. [Lin00] Lindqvist, U. and Porras, P., "eXpert-BSM: A Host-based Intrusion Detection Solution for Sun Solaris", http://www.sdl.sri.com/papers/expertbsm-acsac01/
8. [Lip00] Lippmann, Richard R., Hanes, J., Fried, D., Korba, J., and Das, K. "Analysis of the 1999 DARPA Off-Line Intrusion Detection Evaluation", *Proceedings of DARPA Information Survivability Conference and Exposition*, DISCEX'00, January 25-27, Hilton Head, SC, 2000, Proceedings of *Recent Advances in Intrusion Detection* (*RAID),* Toulouse, France, 2-4 October 2000, Springer-Verlag Lecture Notes in Computer Science, October, 2000, pp. 162-182.
9. [McH01] McHugh, J., quoted during panel discussion at RAID 2001.
10. [NIPC01] National Infrastructure Protection Center Advisory 01-004, http://www.nopc.gov/warnings/assessments/2001/001-004, March 2001.
11. [Pearl88] Pearl, J. (1988) "Probabilistic Reasoning in Intelligent Systems", Morgan-Kaufmann, Menlo Park, CA.
12. [Por97] Porras, P. and Neumann, P. "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances", National Information Security Conference, 1997. http://www.sdl.sri.com/emerald/emerald-niss97.html
13. [Ski98] Skinner, K. and Valdes, A. "EMERALD™ TCP Statistical Analyzer 1998 Evaluation Results", http://www.sdl.sri.com/emerald/98-eval-estat/index.html
14. [SNORT] Roesch, M. http://snort.sourcefire.com/

15. [Val94] Valdes, A. and Anderson, D. (1994) "Statistical Methods for Computer Usage Anomaly Detection using NIDES*", Proceedings of the Third International Conference on Rough Sets and Soft Computing*, San Jose, CA, 10-12 November 1994, pp. 306-311.

16. [Val00] Valdes, A. and Skinner, K. "Adaptive, Model-based Monitoring for Cyber Attack Detection", Proceedings of *Recent Advances in Intrusion Detection* (*RAID),* Toulouse, France, 2-4 October 2000, Springer-Verlag Lecture Notes in Computer Science, October 2000, pp. 80-92.

17. [Val01] Valdes, A. and Skinner, K. "Probabilistic Alert Correlation", Proceedings of *Recent Advances in Intrusion Detection* (*RAID),* Davis, CA, 10-12 October 2001, Springer-Verlag Lecture Notes in Computer Science, October 2001, pp. 54-68.