To appear in *Proceedings of the Second DARPA Information Survivability Conference and Exposition (DISCEX-II)*, June 12-14, 2001, Anaheim, California.

Agile Monitoring for Cyber Defense

Jon Doyle
Laboratory for Computer Science
Massachusetts Institute of Technology
200 Technology Square
Cambridge, MA 02139, USA
doyle@mit.edu

Isaac Kohane
Childrens' Hospital
300 Longwood Avenue
Boston, MA 02115, USA
isaac_kohane@harvard.edu

William Long
Laboratory for Computer Science
Massachusetts Institute of Technology
200 Technology Square
Cambridge, MA 02139, USA
wjl@mit.edu

Howard Shrobe
Artificial Intelligence Laboratory
Massachusetts Institute of Technology
200 Technology Square
Cambridge, MA 02139, USA
hes@ai.mit.edu

Peter Szolovits
Laboratory for Computer Science
Massachusetts Institute of Technology
200 Technology Square
Cambridge, MA 02139, USA
psz@mit.edu

Abstract

The Monitoring, Analysis, and Interpretation Tool Arsenal (MAITA) seeks to support rapid construction and empirical reconfiguration of cyber defense monitoring systems inside the opponent decision cycle through a set of mechanisms including a flexible infrastructure for distributed monitoring processes and signal flows, a monitoring executive that coordinates resource allocation and systemic self-monitoring, and a library of monitoring process types, event descriptions, event recognition methods, alerting decision models, and other forms of monitoring knowledge.

1. Introduction

Effective cyber defense requires substantial monitoring capabilities to track the status of one's own resources and the activities of opponents and third parties. Effective monitoring, in turn, demands the ability to apply correlative and situational knowledge productively and the ability to adapt rapidly to changing circumstances and threats.

Because cyber defense is a broad and adversarial process that will involve many different kinds of knowledge, our effort is to construct a system that supports a broad framework for monitoring activities, with the following characteristics:

- The monitoring system adapts its behavior based on its assessment of circumstances, changing goals, and learning from experience.
- The system innately couples monitoring to response, to assure the integrity of the monitoring system, to make it adaptive, and to support its users in an active defense of the resources being attacked.
- Monitoring and response happen on a large range of time scales, from the recognition of a port scan over a period of seconds to much more subtle subversions that can develop over weeks or months. Very short term activities require complete planning or scripting prior to execution, whereas longer-term actions permit incremental planning in consultation with users and other systems.

- We intend the system to support use by a human analyst. This requires tailoring its knowledge, preferences, strategies, and visual aids to support such human use.

- Effective system management requires basing monitoring and response on explicit knowledge about the situations being monitored, possible objectives of attackers, consequences of various types of compromise, etc. Such explicitness underlies the ability to trade off competing goals when all are not simultaneously achievable, to learn from experience, and to communicate situations, observations, plans and results.

Knowledge enters into effective monitoring in correlating disparate flows of information. Some correlation knowledge works positively in recognizing events of interest by observing temporal trends and by putting together widely-separated indicators of low-visibility events. Other correlation knowledge works negatively to avoid false alarms by checking observed trends against known correlates or consequences. In addition to these types of knowledge about the relationships among objective events, effective monitoring also requires use of knowledge about the subjective preferences and utilities of alerts to recipients.

Addressing rapidly changing circumstances and threats demands the ability to rapidly change the substance and organization of monitoring activities. Some changes may involve specifying only new values for specific monitoring parameters, such as timescales and thresholds. In altering "infocon" levels, the parameter change may entail many consequential alterations, some to other parameters, and others to the organization of monitoring activities. More generally, however, changed threats call for reorganizing or augmenting monitoring activities to watch for events not previously observed, or to expend additional resources and attention to ensuring detection of specific events. Such reorganizations may require adding additional monitoring processes and resources to existing activities. Making such additions within desired reaction times calls for a library of monitor types that defenders can easily instantiate and configure to meet the particular new demands.

Miscreants attempting to break into one site often repeat their attacks at other sites. This provides a good example of the problem of special or temporary concerns. A successful or partially successful attack against one site may call for alerting similar or related sites of the special characteristics of the attack, since the attack may have involved several events which mean little in themselves but which in retrospect appeared essential elements of the successful attack. For example, a random request from an obscure Lilliputian site might immediately precede a seemingly unrelated request from a Blefescucian site in the course of the attack on a particular service. If the succumbing site can alert its neighbors to this fact, the neighbors may repel similar attacks by watching for this specific combination and taking appropriate ameliorating actions pending a longer term solution to the vulnerability. The problem here is how to rapidly modify monitoring systems in place to recognize additional or specialized conditions and how to easily remove these specialized additions as windows of opportunity close or shift.

2 The MAITA system

The MAITA system (Monitoring, Analysis, and Interpretation Tool Arsenal) seeks to provide the means to apply and adapt monitoring knowledge to create and maintain effective monitoring systems. It provides a base architecture for networks of distributed monitoring processes, a monitoring management executive for controlling the monitoring network, a system health monitor for tracking the status of the monitoring network itself, a system of graphical monitoring control panels to permit human operators to observe monitoring results and control monitoring processes, and a library of monitoring knowledge to facilitate reuse of monitoring techniques and solutions.

The MAITA system incorporates several interrelated subsystems and functionalities. Although implementation of these distinct functionalities as separate processes may prove warranted in the future, the current implementation combines the system management executive and health monitoring functionalities into a single process called the "monitor of monitors", or MOM.

We have not intended MAITA to operate as stand-alone or self-contained system, but instead as a portion of the monitoring infrastructure of a larger command and control system that incorporates a diverse set of sensors and systems. MAITA provides computational mechanisms for feeding outputs of existing systems through a network of analytical processes and out to human and automated recipients. Toward this end, it also provides programmatic structures for wrapping existing systems and computational methods to simplify some types of communication and control. It does not provide mechanisms for any type of response other than alert generation, though in work described elsewhere, we use the MAITA mechanisms to inform probabilistic trust models that guide resource allocations aimed at minimizing the impact of system compromises and degradation [25].

3. Monitoring network architecture

The central concept in the MAITA architecture is that of a network of distributed monitoring processes, interacting with and tracking an environment of distributed sensors and processes.

3.1. Signal flows

A typical monitoring process receives one or more "signals" or information streams from sensors or other monitoring processes, and generates streams of reports, alerts, or transformed information.

Each monitoring process communicates information of different types through its own process-specific sets of "input" and "output" terminals. Each terminal transmits or receives a stream of reports of a certain type and meaning. The architecture does not enforce any correspondence between monitoring signals and process terminals, in that a monitoring network may transmit several signals simultaneously through the same process terminal, and may (in rare cases) transmit the same signal to several terminals of a recipient process.

Information flows through the monitoring network by means of connections made between the terminals of monitoring processes. These connections represent a level of abstraction higher than that of internet connections, although internet connections mediate most monitoring process connections. In particular, reports flow through the network by any of an open-ended set of standard communication protocols, currently consisting of persistent ASCII character streams and episodic HTTP connections, but eventually to include SMTP (Simple Mail Transport Protocol) messages, Java RMI (Java Remote Method Invocation), ODBC (Open Database Connectivity), and encrypted versions of these or other protocols. Designers choose which protocol to use by considering the volume, regularity, and type of the information being transmitted. One expects that regular and high-frequency transmissions go through persistent stream connections, while intermittent and low-frequency transmissions go on temporary HTTP or SMTP connections. MAITA decouples use of these protocols to transmit information from particular types of content. Monitoring processes can use HTTP, for example, to transmit plain text content, HTML or XML content, or other alternatives. Standard transmission options provided by the system allow processes or users to request content structured in different formats, as desired.

Each monitor type definition must describe the structure of reports transmitted through its terminals. The structure of these reports rests at a semantic level in terms of concepts meaningful in terms of the information and operations involved in the monitoring process. Each monitor type definition also describes the structure of computational datatype encodings of these reports in structured records called packets (unrelated to internet packets). The MAITA system itself provides automatic means for encoding the data types defined by packets into the various communications protocols supported by connections between MAITA terminals.

MAITA process terminals facilitate interconnection of MAITA processes, but the architecture permits connection of its own processes with external data sources or recipients that lack the specific structure of MAITA terminals as long as those data sources or recipients communicate through one of a number of common protocols, such as HTTP or socket-based streams. One may integrate legacy processes more closely into the monitoring network by enclosing them in wrappers to give them terminal functionality of standard MAITA monitoring processes.

3.2. Processing

To produce its output signals, each monitoring process performs some transformation, analysis or correlation of the information received on its input terminals. Some processes may perform stateless transforms of input signals, while others may exploit process-specific memory, databases, or environmental probes in performing transformation, analysis, or correlation of the input signals.

The MAITA system provides for a wide range of types of processing in two different forms: atomic processes, and composite processes.

Atomic processes perform substantial computations themselves. The MAITA library seeks to include some common statistical and signal-processing processes of this form. The library also includes event descriptions, which one can combine with general event-recognition processes to yield processes that seek to recognize particular types of events. We say more about event recognition methods below. The MAITA architecture includes wrapping code, in Lisp, Java, and eventually other languages, with which one can embed other computational elements, such as existing intrusion detection systems and sensors, in a monitoring network.

Composite processes operate by delegation, feeding data streams through a network of subprocesses, and thus represent an abstract form of the subnetwork that actually does the work. We anticipate use of compilation methods to convert some types of composite processes into atomic processes when efficiency or privacy issues so dictate.

3.3. Control

One exercises control over monitor processes through special bi-directional terminals called "control" terminals. Each monitor process has exactly one control terminal through which it receives and responds to commands from human controllers, the monitoring management executive, or other monitoring processes. The control API consists of instructions falling into several categories, including process status and information requests, connection establishment and disestablishment requests, instructions to start, stop, pause or resume processing inputs, instructions to change processing parameters, and instructions to update control authorization information.

At present, control terminals communicate using the hypertext transport protocol (HTTP), encoding each control API command as an HTTP request. The requests include instructions on the form of the response desired, ranging from no response, plain text response, HTML response, or separate HTTP communications. Future extensions may augment the structure of control terminals to permit the use of the Java remote method invocation and CORBA protocols as alternatives to HTTP.

The function of the monitor process is to respond to commands on its control terminal, and, pursuant to those commands, accept input items on its input terminals and generate output items on its output terminals. The process itself consists of a fluctuating set of computational threads that perform the operations associated with its terminals and connections.

3.4. System security

Effort in developing the MAITA system has initially focussed on realizing the essential structures useful in distributed monitoring. The architecture makes some provision of mechanisms aimed at increasing the security of the monitoring system, but generally has been designed with the expectation that it operates within the protection of more serious security mechanisms.

The current MAITA implementation provides modest security levels through use of a fairly standard Unix-like scheme of user and group passwords and permissions. The architecture also seeks to present a minimal target for port-based attackers by establishing most of its data communications through ephemeral listeners operating out of randomly-assigned ports. Only the system management executive operates at publicized and stable locations. We expect to extend the current communication protocols to encrypted protocols in the future in order to further increase system security.

For more serious efforts at system security, we have begun exploring the notion of *active trust management*, in which a portion of the monitoring effort is used to inform a model of the trustworthiness and reliability of all system elements, which in turn informs the decisions of the system executive in allocating resources to monitoring tasks [25].

4. Monitoring control interface

Use of HTTP for transmission of control operations facilitates system operation through web browsers, using commands entered by hand or through multiple specialized web pages or applets. Such web-based control minimizes requirements for installing specialized software on local machines. The main control panel provided consists of a Java applet that provides a graphical representation of the network of monitoring processes, as depicted in Figure 1. This applet provides means for performing standard control operations on the processes depicted, including calling up process-specific control panels that provide means for modifying the behavior of individual processes when the processes have been designed to permit such runtime control.

5. Monitoring signal displays

Human analysts require the results of analyses to be presented in intelligible forms, such as graphs and charts that convey the important information prominently without dilution by extraneous detail. The MAITA system presently employs several main display types following common forms, but construction of optimized displays has not been a focus of our effort.

MAITA provides *multivariate strip charts* (see Figure 2) of selected streams that display the values of one or more variables on a rectangular graph, with the displayed variables plotted vertically and time plotted horizontally. The system can operate individual strips as well as combination strips in which several individual strip charts are stacked one on top of the other with a common temporal reference on the horizontal axis. The system provides the ability to create combination and multivariate strip charts by selecting various connections or terminals in the process network diagram and then performing the appropriate control-menu operation.

MAITA provides *two-dimensional (2D) maps* (see Figure 3) of variables plotted against each other that display one or more paired variables, with one set of variables plotted on the vertical, and another set plotted on the horizontal. In a 2D map, time does not appear as a dimension of the graph axes. Instead, the temporal window appears only through the number of points plotted; as the temporal window moves, excessively old points are removed from the display, and the new points are added.

Text alerts display sentences, phrases, or words that constitute alerts to the user.

MAITA also supports combinations of these types. For example, one might combine a strip chart with a text alert that states the normality or abnormality of the stream contents being displayed in the strip chart.

We expect to make future extensions that provide additional visual display types, as well as nonverbal audio and synthesized speech alerts.

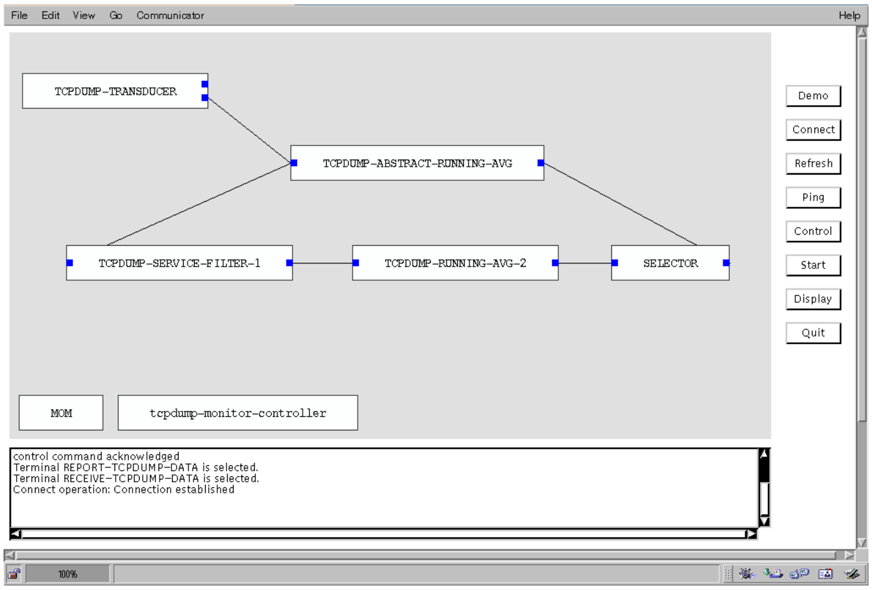

Figure 1. A system control panel, depicting a simple network of monitoring processes. Each labeled rectangle in the shaded area iconifies a monitoring process. Each peripheral square on the left side of a process icon represents an input data terminal; squares on the right represent outputs terminals. In the network depicted here, the Transducer process (upper left) streams TCPdump data to the monitoring processes. The Abstract process (upper center) computes running averages of signal levels by feeding its input stream through the subnetwork of three processes (center). The Mom (lower left) exercises control over the overall system, and the TCPdump controller (lower left) aids in constructing monitoring structures for TCPdump data streams. The rank of control buttons to the right of the network display provide single-click operations on selected processes. The log window at underneath the network display records operations and their results.

6. Monitoring management executive

The MAITA monitoring management executive serves to schedule system management tasks, coordinate execution of these tasks, and allocate resources to these tasks, all with the guidance of human masters.

The present management executive consists of simple mechanisms involving no special intelligence and requiring human controllers to make some decisions. We expect to supersede these simple mechanisms with more uniform mechanisms that incorporate standard plan-execution systems that manage and carry out hierarchically-organized agendas of tasks, accompanied by resource allocation mechanisms using market-based and decision-theoretic methods for obtaining efficient allocations [4]. In our work on active trust management, we seek to exploit such mechanisms to base task execution and resource allocation on estimations of the trustworthiness and reliability of different resources for different tasks [25, 6].

7. System health monitor

We use the term “monitor of monitors” or MOM to highlight the functionality of tracking and maintaining the health or proper operation of the monitoring system itself. The initial implementation of the MOM incorporates several different mechanisms toward this end.

The first element of the basic health monitoring consists of periodic checking of the status of monitoring processes. In this element, the MOM sends status requests to each monitor according to a process-specific schedule. Processes responding with inappropriate responses, and nonresponsive processes, call for further action, either by the MOM or by a human controller. The simplest type of action consists of replacement of the ailing process with a new instance and rerouting of the information flows away from the ailing process to the replacement.

The second element of the basic health monitoring consists of the ability of ailing processes to send distress messages or status warnings directly to the MOM apart from the periodic status checks. The MOM, on receiving such

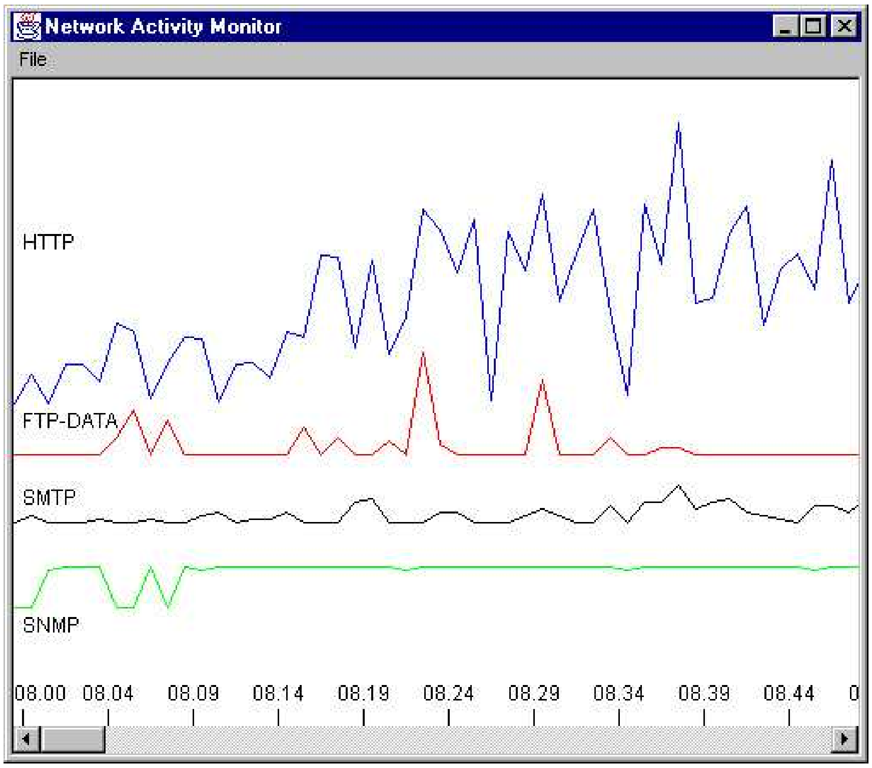

Figure 2. A strip chart displaying several signals in a moving temporal window. The chart displays volumes of various network traffic over time. The display imaged here omits indication of the signal scales.

notices, can take corrective action.

The third element of the basic health monitoring consists of provision of monitors called "grandMOMs" that serve to track the health of the MOM itself. A grandMOM operates as a process itself under the management and protection of a MOM. A single MOM may manage several grandMOMs, and a single grandMOM may track the health of several MOMs. Each grandMOM performs periodic status checks of a MOM, just as a MOM does of it, and also receives reports from other processes of failure to establish contact with or receive responses from a MOM. When one of its MOMs fails, the grandMOM creates a new MOM (a step-MOM) to manage the processes formerly managed by the defunct MOM, and notifies these processes of the replacement of the defunct MOM by the step-MOM.

The fourth element of the basic health monitoring consists of a persistent database recording the information about all processes used by the MOM. When a MOM begins operation, it goes through the database to verify the accuracy of the information stored therein and correct the information where necessary. This involves checking to see that the monitoring processes listed in the database still exist and respond properly to status queries, and that the connections between process terminals listed in the database still exist.

Our work on active trust management seeks to improve on these basic mechanisms with more elaborate evidentiary models of the trustworthiness and functionality of monitoring processes and monitoring resources.

8. Monitoring knowledge library

The MAITA system monitoring knowledge library supports construction of monitoring systems by facilitating creation of new instances of monitor types found useful in the past. These monitor types include both individual monitoring process types and compound monitoring processes containing subnetworks of monitoring processes.

While the library may contain many specific types of monitoring processes, one of the most useful general processes consists of an event recognition process based on matching data streams against a "trend template" that characterizes the temporal and qualitative structure of the type of event in question in terms of relations between different

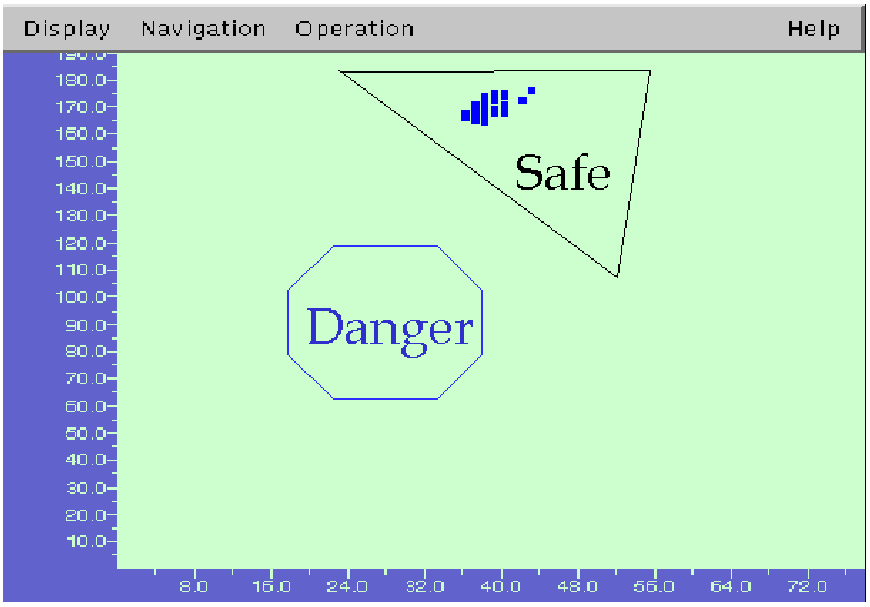

Figure 3. A display plotting one signal against another, annotated with regions of safety and danger. The regions depicted in this diagram serve only to illustrate the capability.

quantities and their changes over different temporal intervals. We discuss trend templates in more detail shortly.

Another useful general process type consists of a filter process that one specializes with an alerting model characterizing the conditions under which alerts should be conveyed to different recipient classes. Alerting models encode knowledge about the likelihood of different classes of alerts or reports, time and other costs of transmission, utility to different recipients or recipient classes, and legal or regulatory constraints and policies. The filter process uses this information to make rational choices about who to tell what, and when and how.

To aid in the construction of these trend-detection and filtering processes, the library also contains sets of trend templates and alerting models.

8.1. Event recognition

A trend template (TT), as introduced in the TrenD_x system [12, 11, 18, 15, 9], is an archetypal pattern of data variation in a related collection of data that serves as a characterization of a type of event. For example, a particular information security trend template might characterize an event consisting of a port sweep followed by increased traffic using some particular port to a small set of destinations rarely seen before.

Each TT has a temporal component and a value component. The temporal component includes landmark time points and intervals. Landmark points represent significant events in the lifetime of the monitored process. They may be uncertain in time, and so are represented with time ranges (min max) expressing the minimal and maximal times between them. Intervals represent periods of the process that are significant interpretation. Intervals consist of begin and end points whose times are declared either as offsets of the form (min max) from a landmark point, or offsets of the form (min max) from another interval's begin or end point. The temporal representation is supported by a temporal utility package (TUP) that propagates temporal bound inferences among related points and intervals [17, 16]. The value component characterizes constraints on individual data values and propositions and on computed trends in time-ordered data, and specifies constraints that must hold among different data streams.

To illustrate the representation, Figure 4 presents portions of a simplified trend template that describes a staged takeover of an FTP site, in which intruders conduct a series of probes aimed at cracking a user password, set up an FTP site for transshipment of unauthorized materials, and then publicize the existence of the site to others, who later begin using the site more and more until the usage saturates the capacity of the host. The trend template contains landmark times (indicated by the `:landmarks` entry) corresponding to initiation of probing, achievement of compromise, initiation of transfers through the site, the point at which the exploit saturates the capabilities of the site, and the current time. We omit constraints characterizing the probing and latency intervals, but characterize the loading period as an interval constrained to exhibit saturating FTP volume and host load averages. The constraint definitions indicate the parameters being constrained (e.g., `FTP-VOLUME`) and the qualitative shape formed by the values of the parameter over the interval. The shape model `(s-curve (d1 +))`, for example, indicates an S-shaped curve connecting two levels, with a positive first derivative in the middle section of the s-curve, that is, an S-curve starting from a low level of FTP volume and leveling off at a higher level. (A simpler model might use a simple linear model `(linear (d1 +))` instead of the s-curve.) Similarly, we characterize the subsequent "continuing" period as an interval constrained to exhibit continued exploitation at saturation levels. We characterize these intervals as consecutive sequential phases of the overall event. The temporal relations express lower and upper bounds on the duration of intervals between time points given in the first two elements of each four-element list. Bounds of "0 0" indicate identification of two points. The relations in this trend template do not bound the duration of probing, and require only small lower bounds on the duration of latency and loading periods, but require longer periods of continuing exploitation to rule out happenstance temporary periods of saturation.

In matching a trend template to data, two tasks are carried out simultaneously. First, the bounds on time intervals mentioned in the TT are refined so that the data best fits the TT. For example, a TT that looks for a linear rise in a numeric parameter followed by its holding steady while another parameter decays exponentially must find the (approximate) time boundary between these two conditions. Its best estimate will minimize deviations from the constraints. Second, an overall measure of the quality of fit is computed from the deviations. The measures of quality that tells how well various TTs fit the monitored data become either time-varying signal or propositional outputs of the signal correlators and trend detectors, and provide the appropriately processed inputs for making monitoring decisions.

We believe that trend templates and associated matching mechanisms provide an approach to event recognition that goes beyond the capabilities standard signature and anomaly methods and their direct combinations. See [5] for an extended discussion.

```
(deftt FTP-TAKEOVER
 :landmarks '(initiation compromise
        exploitation saturation now)
 :intervals
  ((definterval PROBING
        <omitted>)
   (definterval LATENCY
        <omitted>  )
   (definterval LOADING
    :constraints
    ((defconstraint SATURATING-FTP
        :parameters (FTP-VOLUME)
        :model (s-curve (d1 +)))
     (defconstraint SATURATING-LOAD
        :parameters (LOAD-AVERAGE)
        :model (s-curve (d1 +))) ))
   (definterval CONTINUING
    :constraints
    ((defconstraint SATURATED-FTP
        :parameters (FTP-VOLUME)
        :model (linear (d1 0)))
     (defconstraint SATURATED-LOAD
        :parameters (LOAD-AVERAGE)
        :model (linear-curve (d1 0)))
   )))
 :relations
 '((consecutive-phase probing
     latency loading continuing)
   (initiation (begin probing) 0 0)
   (compromise (end probing) 0 0)
   (exploitation (begin loading) 0 0)
   (saturation (end loading) 0 0)
   (now (end continuing) 0 0)
   (compromise exploitation
    (minutes 1) (days 1))
   (exploitation saturation
    (minutes 2) (days 1))
   (saturation now
    (minutes 10) (days 1)) ))
```

Figure 4. Excerpts from a simplified trend template describing behavior characteristic of compromise and exploitation of a FTP site.

8.2. Alerting models

The library of alerting models incorporates both extant procedures for making alerting decisions and methods for convenient specification of utility information. The medical informatics literature contains an unsystematic variety of alerting procedures, with few tied to explicit notions of utility (see, for example, [14, 13, 20, 22, 23]). We do not know of any systematic approach to characterizing alerting models in the intrusion detection literature. In the intrusion detection literature, simple threshold and procedural mechanisms predominate. The EMERALD system [21, 27] employs a sophisticated combination of these approaches. It uses probabilistic models for quantifying degrees of certainty about the occurrence of some event, and permits one sensor to set the alerting thresholds of another. This method appears to make information about alert utility implicit in the probabilistic and sensor-interaction structures, and appears to make little provision for accommodating differing alert utilities for different alert recipients.

Our ongoing construction of alerting models uses explicit utility models to develop a systematic collection of alerting procedures that includes the ones already reported in the literature. The representations here build on qualitative representations of utility information [28, 7, 29, 8], including logical languages that can express generic preferences ("prefer alerting by machine-synthesized telephone calls to sending email through compromised networks"), and that relate this notion of preference to the notion of problem-solving or planning goals (interpreting goals as conditions preferred to their opposites, other things being equal). We are developing utility models that combine both qualitative preference information with approximate numerical models of common utility structures (for example, utility models that increase up to some time and then drop off to model deadline goals, as in [10]), along with automatic procedures for combining such information into qualitative decision procedures and numerical multiattribute utility functions suitable for quick evaluation of alternatives.

Recognition of an event calls for deciding what to do with that information: who to notify, when to notify them, and how to notify them. Tailoring the methods used for making alerting decisions constitutes a key method for making the monitoring system responsive to individual analysts. Most of the effort in guiding alerting decisions consists in describing the utility of different results to different agents in different situations.

Alerting decisions may in general depend on various factors, including the suspected target of the threat, the seriousness of loss or compromise of that target, and the likelihood of success of the current attack. These decisions also depend on the event being reported since analysts have priorities among the conditions of interest to them, and normally wish to hear about the most urgent and important items right away, with the lesser items deferred for consideration later. Alerting decisions depend on the recipient since different analysts will have different interests, priorities, and tasks. Alerting decisions may also depend on the sets of possible recipients and media used to communicate alerts. For example, unreliable transmission or receipt times may call for copying alerts to backup recipients or through alternative transmission media.

9. Experience

The focus of our development of the MAITA system lies in monitoring tasks related to information assurance and survivability, although we have also applied these methods to datasets drawn from battlefield movement analysis and clinical medicine. The MAITA trend detection methods build on clinical trend detection explorations that demonstrated the ability to recognize and differentiate competing clinically significant trends in monitoring patients in intensive care units and pediatric medicine [11, 3, 26]. In battlefield movement detection, our methods identified both significant large and low-visibility episodic movements [1].

In the computing security arena, the bulk of our exploration to date has made use of TCPdump data from the Lincoln Laboratory intrusion detection evaluation data sets. We are currently connecting our monitoring system to live data streams from intrusion detection systems operating in local networks within the MIT Artificial Intelligence Laboratory and the MIT Laboratory for Computer Science.

Although we have obtained descriptions of various attack signatures from intrusion detection systems, our local monitoring focuses on interpreting the reports of IDSs in terms of longer-term trends in traffic levels of various services, as in the FTP-takeover event discussed above. Signature-based detection methods serve to characterize specific sequences, or classes of sequences of concrete events that constitute intrusions, and anomaly-based detection methods serve to identify temporal intervals in which things differ from expectations. The events one can characterize using trend templates appear to go beyond the capabilities of standard signature and anomaly methods and their simple combinations [5].

10. Related work

The MAITA architecture has significant similarities to the architecture used in the EMERALD intrusion detection system [21, 27]. The basic EMERALD system provides a distributed set of monitoring processes or capabilities, organized hierarchically in a way that scales with the size of the enclave being protected. EMERALD monitoring processes have a standard form across all levels, one that combines a set of rules with a set of statistical patterns, and incorporates a uniform method by which one process may subscribe to the results of others. The MAITA architecture provides similar capabilities, though allowing somewhat greater flexibility and expressiveness in prescribing the operation of the processes, and apparently allowing a more flexible set of communication mechanisms. The EMERALD system is engineered to provide basic protections for its own operations at all levels. In contrast, work on the MAITA system has focussed on the structure and content of the monitoring processes with the expectation of piggybacking on protections afforded by environmental system components.

The knowledge embodied by intrusion detection systems tends to focus on fairly low levels of signals. In the EMER-

ALD system, for example, rules tend to be simple and statistical methods tend to be prominent at the lowest levels of the monitoring hierarchy; at higher levels, rules become more complex, while the contributions of statistical methods diminish. The main bodies of knowledge for rules concern signature of varying degrees of complexity, but most represent fairly local considerations, rather than extended temporal and statistical behaviors described in trend templates. The statistical models may be constructed manually or constructed and adapted mechanically, but most are based on fairly gross properties of the system being monitored. Our work on MAITA looks also to developing methods for basing monitors on more complicated statistical and probabilistic models, using Bayesian networks involving terms related to concepts in the situation knowledge base, and exhibiting situational dependence in which the probabilistic network used may itself be changed as the situation changes.

Specific monitoring systems, as opposed to codifications of libraries of knowledge for constructing monitoring systems, are well represented in the literature and in commercial products. The most relevant work, other than our own, on monitoring knowledge and methods appears in the literature on trend detection and "temporal abstraction", especially in the work of Shahar [24] and Das [2] at Stanford. These efforts focus on representing temporal relationships and on methods for identifying patterns of temporal relationships as instances of more abstract events. This work provides a good foundation for monitoring, but intelligent monitoring and analysis involves more than just temporal information. Structuring relevant sorts of non-temporal information, especially information about logical implication, statistical correlations, and causation, is crucial, but lacking in most abstraction-based treatments. Statistical trend detection, on the other hand, does not adequately exploit the constraints and structuring information that templates provide. Our representations for monitoring conditions seek to integrate the best representations devised for each of these separate types of knowledge.

The Guardian project [14, 13] at Stanford has developed a highly dynamic programming environment for the construction of very flexible monitoring systems. It puts very strong emphasis on giving the system the ability to reason, during the monitoring process, about the most appropriate data collection, interpretation and integration strategies. It places correspondingly less emphasis on the ease of constructing relatively simpler monitoring strategies beforehand, and has not developed detailed libraries of monitoring modules to support easy assembly. Our work on the MAITA system attempts to make explicit use of background knowledge about monitoring and about the domain and monitoring task more at the time a monitoring process is assembled and configured than dynamically during its execution. We believe that this approach leads to more efficient monitoring systems and greater ease of their development and configuration.

Commercial technology for monitoring and control offers good models of some of the capabilities we seek, but does not offer the flexibility, modularity, or construction tools of interest here, so we have developed MAITA without relying on such existing systems. The G2 system [19] offered by Gensym Corporation provides an example. This system provides a good base of the "object-level" monitoring capabilities, namely the ability to accept inputs from several types of sources, a library of single-signal filters (linear extrapolation, fourier transforms, etc.), and a knowledge-based reasoning component for constructing multisignal analysis systems. While the library of single-signal filters and the primitives of the multisignal analysis language provide good starting points, they fail to cover some important types of knowledge (probabilities, causality). More importantly, G2 provides only a programming language, and not a structured library of procedures at various levels of specificity. Finally, G2 is structured as a heavyweight, stand-alone application, and does not provide the environment needed to support distributed efforts by multiple collaborating analysts. In G2, adding a new process to monitor some additional threat requires programming the new recognition procedure (without library support) and then recompiling and reinstalling the resulting overall monitoring process. Distributed and collaborative monitoring efforts instead demand the ability to toss a new monitoring element into an ongoing process, as we have provided in the MAITA system.

11. Conclusion

The monitoring process infrastructure, executive, self-maintenance, and knowledge library provided in the MAITA system constitutes a flexible set of mechanisms with which to construct, operate, and adapt monitoring systems in information assurance and survivability applications.

12. Acknowledgment

We thank the referees for valuable comments and suggestions. This work is supported by DARPA under contract F30602-99-1-0509.

References

[1] P. Cohen, R. Schrag, E. Jones, A. Pease, A. Lin, B. Starr, D. Gunning, and M. Burke. The darpa high-performance knowledge bases project. *AI Magazine*, 19(4):25–49, Winter 1998.

[2] A. K. Das and M. A. Musen. A comparison of the temporal expressiveness of three database query methods. In *Nineteenth Annual Symposium on Computer Applications in Medical Care*, pages 331–337, New Orleans, LA, 1995.

[3] M. T. DeSouza. Automated medical trend detection. Master's thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 2000.

[4] J. Doyle. A reasoning economy for planning and replanning. In M. H. Burstein, editor, *Proceedings of the 1994 ARPA/Rome Laboratory Knowledge-Based Planning and Scheduling Initiative Workshop*, pages 35–43, San Francisco, CA, 1994. Morgan Kaufmann.

[5] J. Doyle, I. Kohane, W. Long, H. Shrobe, and P. Szolovits. Event recognition beyond signature and anomaly. In *Proceedings of the Second IEEE SMC Information Assurance Workshop*. IEEE, IEEE Computer Society, June 2001.

[6] J. Doyle and M. McGeachie. Qualitative preferential control of alerting behavior. Submitted for publication, 2001.

[7] J. Doyle, Y. Shoham, and M. P. Wellman. A logic of relative desire (preliminary report). In Z. W. Ras and M. Zemankova, editors, *Methodologies for Intelligent Systems, 6*, volume 542 of *Lecture Notes in Artificial Intelligence*, pages 16–31, Berlin, Oct. 1991. Springer-Verlag.

[8] J. Doyle and M. P. Wellman. Representing preferences as *ceteris paribus* comparatives. In S. Hanks, S. Russell, and M. P. Wellman, editors, *Proceedings of the AAAI Spring Symposium on Decision-Theoretic Planning*, 1994.

[9] J. Fackler, I. J. Haimowitz, and I. S. Kohane. Knowledge-based data display using $\mathrm{TrenD_x}$. In *AAAI Spring Symposium: Interpreting Clinical Data*, Palo Alto, 1994. AAAI Press.

[10] P. Haddawy and S. Hanks. Representations for decision-theoretic planning: Utility functions for deadline goals. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 71–82, San Mateo, CA, 1992. Morgan Kaufmann.

[11] I. J. Haimowitz and I. S. Kohane. Automated trend detection with alternate temporal hypotheses. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 146–151, Chambery, France, 1993.

[12] I. J. Haimowitz and I. S. Kohane. An epistemology for clinically significant trends. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 176–181, Washington, DC, 1993.

[13] B. Hayes-Roth, S. Uckun, J. E. Larsson, J. Drakopoulos, D. Gaba, J. Barr, and J. Chien. Guardian: An experimental system for intelligent ICU monitoring. In *Symposium on Computer Applications in Medical Care*, Washington, DC, 1994.

[14] B. Hayes-Roth, R. Washington, D. Ash, R. Hewett, A. Collinot, A. Vina, and A. Seiver. Guardian: A prototype intelligent agent for intensive-care monitoring. *Journal of AI in Medicine*, 4:165–185, 1992.

[15] I. Kohane and I. Haimowitz. Hypothesis-driven data abstraction. In *Symposium on Computer Applications in Medical Care*, Washington, DC, 1993.

[16] I. S. Kohane. Temporal reasoning in medical expert systems. In R. Salamon, B. Blum, and M. Jørgensen, editors, *MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics*, pages 170–174, Washington, Oct. 1986. North-Holland.

[17] I. S. Kohane. Temporal reasoning in medical expert systems. TR 389, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, Apr. 1987.

[18] I. S. Kohane and I. J. Haimowitz. Encoding patterns of growth to automate detection and diagnosis of abnormal growth patterns. *Pediatric Research*, 33:119A, 1993.

[19] R. Moore, P. Lindenfilzer, L. B. Hawkinson, and B. Matthews. Process control with the g2 real-time expert system. In *IEA/AIE '88: Proceedings of the First International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, volume 1, pages 492–497, Tullahoma, TN, June 1988. ACM.

[20] S. M. Ornstein, D. R. Garr, R. G. Jenkins, P. F. Rust, and A. Arnon. Computer-generated physician and patient reminders. *Journal of Family Practice*, 32:82–90, 1991.

[21] P. A. Porras and P. G. Neumann. Emerald: Event monitoring enabling responses to anomalous live disturbances. In *Proceedings of the Twentieth National Information Systems Security Conference*, Oct. 1997.

[22] D. M. Rind, C. Safran, R. S. Phillips, W. V. Slack, D. R. Calkins, T. L. Delbanco, and H. L. Bleich. The effect of computer-based reminders on the management of hospitalized patients with worsening renal function. In P. Claytons, editor, *Proceedings Symposium Computer Applications in Medical Care*, pages 28–32, Washington, DC, 1991. McGraw-Hill.

[23] D. M. Rind, C. Safran, R. S. Phillips, Q. Wang, D. R. Calkins, T. L. Delbanco, H. L. Bleich, and W. V. Slack. Effect of computer-based alerts on the treatment and outcomes of hospitalized patients. *Archives of Internal Medicine*, 154:1511–1517, 1994.

[24] Y. Shahar. *A Knowledge-Based Method for Temporal Abstraction of Clinical Data*. PhD thesis, Stanford University, Stanford, CA, 1994. Available as Computer Science Department report CS-TR-94-1529.

[25] H. Shrobe, J. Doyle, and P. Szolovits. Active trust management for autonomous adaptive survivable systems. Technical report, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 2000.

[26] C. L. Tsien. *TrendFinder: Automated Detection of Alarmable Trends*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, April 2000.

[27] A. Valdes and K. Skinner. Adaptive, model-based monitoring for cyber attack detection. In *Recent Advances in Intrusion Detection (RAID 2000)*, Toulouse, France, Oct. 2000.

[28] M. P. Wellman and J. Doyle. Preferential semantics for goals. In *Proceedings of the National Conference on Artificial Intelligence*, pages 698–703, 1991.

[29] M. P. Wellman and J. Doyle. Modular utility representation for decision-theoretic planning. In *Proceedings of the First International Conference on AI Planning Systems*, 1992.