Event Recognition Beyond Signature and Anomaly

Jon Doyle, Isaac Kohane, William Long, Howard Shrobe, and Peter Szolovits

Abstract— **Notions of signature and anomaly have formed the basis of useful methods in cyber defense, but even in combination provide only weak evidence for recognizing many events of interest. One can recognize many important events without requiring signatures of specific ways the events can take place and without treating every anomalous behavior as an event. We describe an approach to event recognition that subsumes and extends signature and anomaly methods by starting from a richer language for characterizing events. This paper explains how recognition methods based on this richer event-characterization language offer means for overcoming the limitations constraining signature and anomaly methods.**

I. Signature and anomaly methods

NOTIONS of signature and anomaly form the backbone of current methods for intrusion detection and cyber defense.

Signature-based methods examine the operations performed by processes or requests received by hosts to find a sequence of operations or requests that match a specified pattern. Matching patterns of operations that constitute illegitimate access provides evidence for an attack having taken place. Sometimes failing to match expected patterns of legal operations may also signal an attack.

Anomaly-based methods compare usage statistics for some current period against statistical norms developed from previous periods or other considerations. Periods for which the statistics vary too much from the norms constitute anomalies. Anomalous behavior, in turn, constitutes evidence, though possibly weak evidence, for an attack having taken place.

A. Strengths and weaknesses

While both signature and anomaly methods provide evidence to guide intrusion assessments, their most common forms suffer limitations in the quality of the evidence they can provide.

Signature methods are too special. Attackers can use many different patterns of operations to achieve the same end. For example, an attack might involve conducting several concurrent trains of operations. When the attacker can vary freely the interleaving of the steps of these concurrent trains of operations, a defender depending on sequence recognition must stand ready to recognize a large number of possible sequences as instances of the attack. In other cases, the attackers can expand the variety of alternative methods indefinitely by introducing additional detours in the main attack sequence, so going far beyond the multiplicity involved in mere interleaving. More importantly, the defender always has interest in identifying the compromise that has occurred, for responding to the compromise in an effective way requires such knowledge. Apart from law enforcement, however, the defender generally cares less about correctly identifying the particular method used to effect the compromise. Correct identification of the attack method can prove helpful in warning other sites, but such warnings might hinder defense when attackers vary their methods from site to site.

Anomaly methods are too general. The most dangerous attackers generally know how to hide their activities in the noise, working through scattered low-visibility events that defenders cannot recognize apart from specific expectations about what the attackers might do, expectations that step outside the conception of anomaly as nonspecific recognition of something outside the ordinary. Further, one recognizes an anomaly only with respect to some baseline periods of normality, making recognition difficult for events that do not fall completely within a comparable interval. This possibility increases in likelihood when attackers can make reasonable guesses about the periods of comparison and the nature of normal behavior. More fundamentally, anomaly encompasses anything apart from the norm. It does not distinguish intentional compromise from benign changes in the external environment or inexplicable accident, and so increases the rate of false alarm.

Both signature and anomaly methods offer strengths and weaknesses, but neither's strength fully compensates for the weakness of the other. For example, FTP traffic growing from normal levels to a saturation level and staying there signals a familiar class of intrusion, but no conjunction of signature and anomaly methods characterize this class. None of the FTP transactions constitutes an attack in itself, and indeed, many may be ordinary and legitimate transactions. The traffic levels certainly count as anomalous, but so do vast numbers of other patterns representing mere accident or very different compromises.

B. Illustrating the weaknesses

To better understand the pragmatic weaknesses of signature and anomaly methods, consider the following fictitious description of events that amalgamates aspects of several actual attacks suffered by the MIT Artificial Intelligence Laboratory and Laboratory for Computer Science in recent years.

A lab ensemble of computers runs a visual surveillance and monitoring (VSM) application. On January 12, 2002 several of the machines experience unusual traffic from out-

Jon Doyle, William Long, and Peter Szolovits are with the Massachusetts Institute of Technology Laboratory for Computer Science, Cambridge, Massachusetts 02139, USA. E-mail: {doyle,wjl,psz}@mit.edu.

Isaac Kohane is with Childrens' Hospital, Boston, Massachusetts 02115, USA. E-mail: isaac_kohane@harvard.edu.

Howard Shrobe is with the Massachusetts Institute of Technology Artificial Intelligence Laboratory, Cambridge, Massachusetts 02139, USA. E-mail: hes@ai.mit.edu

side the lab. Intrusion detection systems report observing several password scans. Fortunately, after about 3 days of varying levels of such activity, things seem to return to normal, and remain apparently normal for another 3 weeks.

At that time, however, a machine named Harding that serves crucial functions within the VSM application begins to experience unusually high load averages. The application components running on this machine begin to receive less than the expected quality of service. The load average, degradation of service, the consumption of disk space and the amount of traffic to and from unknown outside machines continue to increase to annoying levels, but then level off.

On March 2, a machine named Grant crashes. Fortunately, the VSM application was designed to adapt to unusual circumstances. The application considers whether it should migrate the computations which would normally have run on Grant to run on Harding instead. It considers these computations critical to the application, but decides despite the odd circumstances noticed earlier on Harding that the migration seems reasonable.

Why did the application consider the migration to Harding reasonable in spite of clearly anomalous behavior? It did so because it recognized the events on Harding as a case of someone guessing a user password and setting up an unauthorized FTP site for transshipment of files, such as illicit software or images. The load on the server increased as word spread about this new transshipment site, and leveled off as demand saturated the machine. The observed events provided no evidence that the root account had been compromised, so the application had little reason to worry that critical computations migrated to Harding would experience any further compromise. The system needed to run those computations somewhere. Even though Harding was loaded more heavily than expected, it still represented the best pool of available computational resources; other machines were even more heavily loaded with other critical computations of the application.

This scenario illustrates in greater detail the inadequacy of signature and anomaly methods in guiding recognition of and response to the intrusion. The intrusion detection systems certainly noticed worrying events in the early stages of the scenario, and detected these events with signature and anomaly methods. Such methods also would classify the high levels of FTP traffic as anomalous, and possibly also so classify some of the individual FTP transactions based on user history. Mere classification as anomaly, however, would not provide the VSM application with the specificity needed to effect the transfer of critical processes to Harding, as the class of anomalous behavior also includes hostile possession of root passwords and the like.

Now consider instead a somewhat different scenario in which the VSM application serves in protecting a threatened diplomatic mission during a period of international tension. In this scenario, the intrusion detection systems, as before, observe a variety of information attacks being aimed at Harding, but now with at least some of these attacks of a type known to occasionally provide root access to a machine like Harding. A period of no anomalous behavior other than a periodic low volume communication with an unknown outside host follows these attacks on Harding. This time, when Grant crashes, the VSM application decides against using Harding as the backup. The event recognized admits significant probability that an intruder gained root access to Harding. The setting of international tensions suggests some probability of malicious political intent behind the intrusion. The periodic communication with the unknown outside host bears some probability of representing attempts to contact an outside control source for a "go signal" initiating serious spoofing of the application. Under these circumstance, the VSM application chooses to shift the computations to a different machine in the ensemble even though it is considerably more overloaded than Harding.

C. *Transcending signature and anomaly*

These differing scenarios show how effective response to intrusion requires transcending the capabilities offered by straightforward signature and anomaly methods.

Recognition of the FTP-site hijacking pattern requires melding the ideas of sequences and statistics in a very different way than one can obtain with parallel operation of sequence and anomaly detectors. The hijacking implies a correlation between the history of subevents that match standard intrusion-detection signatures and a specific pattern of anomalous behaviors, in which the magnitudes of the anomalies fit an increasing and saturating curve (an S-curve) to some degree of accuracy.

In this recognition, mere observation that the increasing load levels constitute anomalies in each time slice does not support recognition of the hijacking event, because a succession of anomalous intervals just add up to a longer anomalous interval. Anomaly detection then might provide a report for the longer interval of the same form as for the shorter intervals. Such reports might suffice to attract human attention, but do not suffice to distinguish the hijacking event from other intrusions. At the same time, recognition of the hijacking proceeds from observing that the load levels increase to saturation, irrespective of any observation that the load levels during this process constitute anomalies.

One might seek to view detection of the increasing and saturating load in terms of an abstract signature over interval statistics. This moves in the right direction, going significantly beyond simple combination of signature and anomaly detection. Even so, to obtain the desired recognition one must expand the conception of signature beyond the traditional notion of sequences or patterns expressed in regular expressions or context-free languages. Recognition of the FTP hijacking might not depend much on the shape of the increasing and saturating curve of load levels, but recognition of other events can require distinguishing linear, quadratic, or exponential curves, or even different slopes of linear curves. For example, one might expect linearly increasing growth of certain traffic flows in organizations that linearly grow in size each year, and wish to distinguish such secular growth in traffic from exponential increases characteristic of cascaded attacks or exploitations in which each perpetrator triggers several more.

For these reasons, we believe that in order to guide effective response to intrusion and compromise, event recognition must go beyond signature and anomaly to recognize patterns more abstract than signatures but more specific to the particular events of interest than anomaly. These intermediate levels in essence taxonomize different degrees of sensitivity and specificity with respect to different types of events. In any particular diagnostic reasoner, it is well known that one must trade off the sensitivity of recognizing true events against specificity (the probability that *only* true events are recognized). That trade-off is often shown in an ROC (Receiver Operator Characteristics) curve, whose general form shows that for any degree of increase in either sensitivity or specificity, the other decreases. An improved diagnostic method is one that reduces the magnitude of this reciprocity. Often an enriched representation, which makes it possible to distinguish previously-unrecognizable situations, helps to improve ROC performance. This is the intuition behind our present work.

The following discusses methods by which one can recognize many important events without requiring signatures of specific ways the events can take place and without treating every anomalous behavior as an event.

II. Characterizing events

We seek to characterize different types of events in linguistic descriptions. We use such descriptions to tell the recognition system which events interest us. The recognition system, in turn, uses the descriptions both in recognition processes that match descriptions to sensor information and in explanation processes that communicate and justify recognized instances.

To understand better what expressions the event description language should encompass, consider again the FTP hijacking scenario presented above. One can describe the pattern of activity resulting from the establishment of the FTP transshipment site in terms of activities during several temporal regions. First there was a period of attacks (particularly password scans). Then there was a "quiescent" period. Then there was a period of increasing degradation of service. Finally, there was a leveling off of the degradation but at the saturation level.

One can resolve this summary into finer details that give more insight and perhaps improve the likelihood of recognition. To do this, we describe the trends of average resource load levels and the average volume of traffic from external sites. During the initial attack and quiescence periods, the load levels stay roughly constant while the external site activity goes up and down, because the attacks themselves do not involve much effort. During the exploitation and saturation phase, the load average climbs to saturation well before the external site activity levels out, because a few initial misusers suffice to swamp the host while word of the site continues to spread to further misusers.

One can also describe the pattern of activity involved in the embassy surveillance scenario in terms of activities during several temporal regions, coupled with environmental information. First there was a period of attacks seemingly aimed at obtaining root access occurring during a period of heightened international tension related to the application being run. Then there was a "false peace" period of no attacks (or merely normal attacks) coupled with periodic low-volume foreign communications.

To permit expression of such characterizations of the scenarios, we seek to enrich the event description language. Our starting point in this enrichment follows Haimowitz and Kohane [1], [2], [3], [4], [5], who developed a language of "trend templates" we will call TTL for expressing temporal patterns like those involved in the examples, along with methods for recognizing instances of trend templates in the stream. The key elements of TTL are as follows:

- *Landmark times.* Landmark times represent significant points in the unfolding of the event over time, such as boundaries between different phases of the event. These can be concrete times (i.e., fully-specified points on the calendar), but often represent abstract times characterized only by uncertain relations to other time points. The original TTL provided only for simple forms of temporal uncertainty, representing relations of landmark times to other times with time ranges expressing the minimal and maximal times between them.
- *Temporal intervals.* Intervals represent periods of the process that characterize significant subevents. Intervals can bear specific or abstract durations constrained by relations to other intervals and to landmark times. Such constraints on temporal position take a form similar to temporal relations among landmark times. Each interval has beginning and ending times. Relational constraints declare these times as satisfying either uncertain (min max) offsets from other landmark points, or uncertain offsets from another interval's beginning or ending points.
- *Temporal relations.* These relations provide shorthand means for expressing relations among intervals that one could express at greater length using the (min max) relations. The provided interval relations include the Allen [6] interval relations and others.
- *State constraints.* These specify characteristics of objects during temporal intervals, such as constant values, increasing or decreasing values, shapes of curves, etc.
- *Regression functions.* These model criteria for matching templates against data, and so describe means for deciding when events occur when uncertainty exists about starting and ending times.

Our current work on the MAITA system [7] seeks to extend the original trend template language in several ways, including augmenting range expressions for indicating temporal relations with more general probability distributions over the occurrence of landmark times. The current constraint language consists mainly of linear and quadratic regression models for numeric data, absolute and relative numerical constraints on functions of the data, and logical combinations of such descriptions and propositions. We expect to augment the state constraint language with probability distributions and additional commonly useful shapes of curves.

To illustrate the representation, Figure 1 presents portions of a simplified trend template that describes the FTP hijacking event. 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 co-occurrence 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.

```
(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)) ))
```

Fig. 1. Excerpts from a simplified trend template describing behavior characteristic of compromise and exploitation of a FTP site.

TrenD_x provides recognition algorithms that employ a partial-match strategy operating over a set of trend templates, each of which consists primarily of temporal constraints characterizing some temporal event. In matching a trend template to data, TrenD_x carries out two tasks simultaneously. First, it refines the bounds on time intervals mentioned in the TT so that the data best fits the TT. For example, in matching a TT that looks for a linear rise in a numeric parameter followed by its holding steady while another parameter decays exponentially, the matcher must find the (approximate) time boundary between these two conditions. Its best estimate will minimize deviations from the constraints. Second, the matcher computes an overall measure of the quality of fit from the deviations.

The trend matching algorithms rely on the Temporal Utility Package (TUP) [8], [9] that propagates temporal bound inferences among related points and intervals.

III. Some comparisons

The preceding examples illustrate some of the limitations that current attack recognition languages appear to suffer, but different extant languages exhibit different limitations. In this section, we examine the apparent limitations of several representative systems.

A. STATL

STATL [10] constitutes one of the most clearly defined languages designed for use in attack recognition. STATL is an extensible language, and the comments that follow refer to the unextended language. Some proper extensions might not have the limitations we attribute to it here.

STATL's strength lies in using familiar programming-language constructs to describe sequential, conditional, and iterative events. These constructs directly address the structure of many automated attacks, since the attacks result from execution of similarly-structured programs.

These strengths do not extend to representing uncertain times and durations, as STATL makes no provision for abstract times or temporal intervals except implicitly in iterative expressions. It instead appears to provide only for concrete times and durations, and does not provide any easy or obvious way of expressing or relating abstract intervals or for expressing or grading uncertainties about when events start and end. STATL seems to tie down all events with specific times and durations, and equates all low-level events with changes in system states. Its only means to represent ambiguity appears to be a transition that may or may not have taken place, whereafter a recognized future event can reset the recognizer to the state before the ambiguous transition.

STATL's expressive limitations regarding times and intervals pose problems in describing the scenarios. Recognizing the FTP and embassy attacks involves characterizations of patterns that refer to abstract times and durations and to relations between abstract temporal intervals. Representing events as transitions through a single chain of states precludes recognizing the achievement of a set of attack preconditions that have no innate required time order. Moreover, the examples highlight the substantial uncertainties involved in exactly when the component events occur. The initial period of increased attack levels in the FTP hijacking, for instance, represents a rise of attack volume above a fluctuating background level of "normal" attacks. One might come to some fairly definite identifications of this event in a forensic analysis well after the events have played out, but attempts to recognize the attack in progress will likely suffer significant doubt about just where the rise in attacks starts and ends.

The patterns illustrated in the scenarios also refer to changes in statistical trends over the intervals in question, such as increasing, decreasing, or constant values. It is not clear that these trends find easy expression in STATL, though perhaps the language-extension mechanism provides the means to include derivatives and other mathematical operations on signals. Similarly, STATL lacks a way of talking about periodic signals, unless perhaps by talking about signals with a concrete period, or by means of a further extension.

Finally, the scenario patterns also refer to causal relationships between events, or more generally, to nonstatistical relationships between events such as intentionality. STATL makes no provision for such relationships between subevents that combine to make a larger event.

B. CISL

CISL [11] represents an important effort to provide a reporting language usable by a variety of intrusion detection systems. CISL's design as a reporting language did not intend to support recognition. However, one can expect languages for describing events in reports to have at least some commonality with languages for describing events to recognition systems. We therefore briefly comment on the utility of the constructs provided by CISL for use in recognition. See [12] for comments on the utility of CISL for event reporting.

CISL shares with STATL a focus on representing concrete times and intervals. It appears to go beyond STATL in providing a somewhat wider range of logical combinations of subevents, perhaps not surprisingly since CISL bears no burden of executing the combinations in the process of recognition. The most prominent extensions related to recognition come in CISL's provision of skeletal constructs for reporting causation and intent. These constructs lack defined meaning, and do not seem adequate even to some purposes of reporting (see [12], [13]). The constructs nevertheless move in the right direction, because recognition methods based on inferring large-scale plans from many piecemeal intentional actions require inclusion of such relations in an attack language.

C. P-BEST

The Emerald [14] intrusion detection architecture goes further than some in seeking to move beyond simple signature and anomaly, by providing for a network of sensors and correlation processes. Emerald employs the P-BEST language [15] for recognition and correlation, but this language really consists of a formalism for expressing probabilistic and linguistic rules, and lacks any concepts specific to event recognition. Probabilistic relationships among concepts certainly play a role in recognition, but the general rule-based framework of P-BEST provides merely a different scheme for general-purpose computation.

For each trend template, one might construct a set of P-BEST rules that derive all possible matchings against data, but this rule set would lack the control needed to identify a single best matching template. The rule set would also provides a poor representation of the event, which has structure independent of any particular recognition mechanism.

D. CYCL

CYCL, the language employed in the CYC system for knowledge representation and reasoning [16], constitutes the most general and expressive language considered here by a wide margin. It provides rather powerful constructs for expressing abstract and concrete concepts across many domains, including logical descriptions, nonmonotonic descriptions, modalities, epistemic and intentional constructs, and, in the MELD extension, probabilistic constructs. The standard ontology provided with CYC contains concepts of time points and intervals, processes, and the like. We expect one can express any of the relationships we seek to capture in a trend template language in CYCL.

That said, CYCL itself was not designed with event recognition as a specific aim, and might prove more general than necessary, with this very generality lessening the specific guidance it provides the designer of event recognition systems. For example, one can use CYCL to describe events in terms of number-theoretic patterns, but this would provide no effective algorithm for recognizing such an event. One hopes the constructs of a recognition language all carry obvious roles in performing event recognition tasks.

IV. Expressivity examples

We now examine some examples of trend descriptions, mostly motivated by hypothetical command and control scenarios, that exercise or exceed the expressive powers of STATL and, in some of the later cases, the original TT language.

Trend templates characterize events that cannot be similarly characterized in languages that provide for descriptions only in terms of specific times and interval durations. Specifically, it appears that one cannot use only concrete times and durations to express descriptions like the following:

A. An X event occurs during a Y event.

B. An X event follows a Y event, and the switch-over time occurred some time between 5 and 7 PM.

C. An X event overlaps and follows a Y event, with the overlap lasting at least 5 minutes.

Correlation of simultaneous event histories also highlights potential difficulties. STATL appears to focus on decomposition of histories into concatenated intervals during which states are constant. There may be a way to describe the multiple overlapping time-varying state constraints expressible in the TT language in these attack languages, but even if that is possible, it is not likely to be convenient given the sequential focus of the languages. Descriptions like the following provide targets for expression here:

D. The resource load activity stayed constant while the external site activity rose and fell, and then the resource load activity rose swiftly to saturation levels while the external site activity rose more gradually and saturated later.

E. The traffic volume through X has been increasing while the traffic volume through Y has been steady.

TTL does not necessarily cover all the concepts desirable in a robust attack language. In particular, it has no facility for expressing probabilistic information. One can easily think of event recognition methods needing to refer to such information. Consider, for example, the following requests a commander might make of a threat-detection system.

F. Warn me if the probability of a class X attack in sector Y goes over 25%.

G. Warn me if the probability of a class X attack in sector Y increases by more than 25% per hour over a period of three hours or more.

H. Discount any threat recognizer that reports attack probabilities that vary too much and too quickly over several five-minute periods.

Similarly, TTL made no explicit provision for expressing negative information (information about absence of events), nor for expressing information about the value of events. Examples here include:

I. No attacks are hitting target X or targets of class Y.

J. The attacks are increasingly on more important targets.

The original TTL also did not include a very rich language for describing waveforms or periodicity. Examples here include:

K. The external requests are oscillating at 1 Hz, with oscillation between larger and smaller volumes of requests taking the shape of a square wave (or sawtooth wave, etc.).

L. The frequency of attacks for which success possibly compromises the secrecy of our plan database is increasing.

M. The frequency of congestion (oscillation) has been increasing for the past day.

Neither STATL nor TTL provide any way of keying recognition methods to the systemic properties of recognition subsystems. Examples of such expressions include:

O. The attacks on command resources are increasing.

P. The success rate of attacks has been decreasing.

Q. It is becoming harder to detect attacks; we are detecting fewer, even though traffic is up without any change in our own behavior.

R. Sensor X is operating at selectivity Y and sensitivity Z on its ROC curve.

S. The effectiveness of our defenses is decreasing; the fraction of attack attempts that cause compromises is increasing.

More generally, as noted earlier, attack recognition languages require means to key methods to intentions and other psychological properties of adversaries. Examples here include:

T. The intent of attack X is Y.

U. The attack hits some machine in every enclave, but appears to prefer Windows NT hosts when they exist.

V. Conclusion

We maintain that some limitations experienced with signature and statistical methods stem not from the methods themselves but from reliance on inadequately expressive languages for describing significant patterns. A richer language, particularly one based on multilevel abstractions and mechanisms for expressing uncertainty in characterizing events, permits one to express more of the central and essential regularities and worrying abnormalities needed to analyze behavior properly. Such a language can increase the difficulty of evading signature or anomaly detection by restructuring the spaces of events in ways that lessen the likelihood that evasive attempts succeed.

We believe that the principal value of an event recognition language inheres in the set of descriptive and analytical concepts it provides right from the start. Such was the advantage of Fortran over early assembler languages for mathematical programming, and such should be the aim of event recognition languages. Thus even when the language provides an extension mechanism, the core language should exhibit richness sufficient to express the structure of many of the events described in the preceding.

Acknowledgment

This work is supported by DARPA under contract F30602-99-1-0509. We thank the referee for helpful suggestions.

References

[1] Ira J. Haimowitz and Isaac S. Kohane, "An epistemology for clinically significant trends," in *Proceedings of the Eleventh National Conference on Artificial Intelligence*, Washington, DC, 1993, pp. 176–181.

[2] Ira J. Haimowitz and Isaac S. Kohane, "Automated trend detection with alternate temporal hypotheses," in *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, Chambery, France, 1993, pp. 146–151.
[3] I. S. Kohane and I. J. Haimowitz, "Encoding patterns of growth to automate detection and diagnosis of abnormal growth patterns," *Pediatric Research*, vol. 33, pp. 119A, 1993.
[4] I. Kohane and I. Haimowitz, "Hypothesis-driven data abstraction," in *Symposium on Computer Applications in Medical Care*, Washington, DC, 1993.
[5] 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.
[6] James F. Allen, "Maintaining knowledge about temporal intervals," *Communications of the ACM*, vol. 26, no. 11, pp. 832–843, Nov. 1983.
[7] Jon Doyle, Isaac Kohane, William Long, Howard Shrobe, and Peter Szolovits, "Agile monitoring for cyber defense," in *Proceedings of the Second DARPA Information Security Conference and Exhibition (DISCEX-II)*. IEEE, May 2001, IEEE Computer Society.
[8] Isaac 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.
[9] Isaac S. Kohane, "Temporal reasoning in medical expert systems," in *MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics*, R. Salamon, B. Blum, and M. Jørgensen, Eds., Washington, Oct. 1986, pp. 170–174, North-Holland.
[10] Steve T. Eckmann, Giovanni Vigna, and Richard A. Kemmerer, "STATL: An attack language for state-based intrusion detection," Dept. of Computer Science, University of California, Santa Barbara, 2000.
[11] Rich Feiertag, Cliff Kahn, Phil Porras, Stuart Schnackenberg, Dan Staniford-Chen, and Brian (editor) Tung, "A common intrusion specification language (CISL)," Tech. Rep., www.gidos.org, 2000.
[12] Jon Doyle, "Some representational limitations of the common intrusion specification language," Tech. Rep., Laboratory for Computer Science, Massachusetts Institute of Technology, October 1999, http://www.medg.lcs.mit.edu/projects/maita /documents/cc2/cisl/revised.txt.
[13] Peter Szolovits, "Detectors should be characterized by likelihood ratios, not posterior probabilities," Tech. Rep., Laboratory for Computer Science, Massachusetts Institute of Technology, June 1999, Revised February 1, 2000. http://www.medg.lcs.mit.edu/projects/maita/ documents/likelihood/detector-likelihoods.pdf.
[14] Phillip A. Porras and Peter G. Neumann, "Emerald: Event monitoring enabling responses to anomalous live disturbances," in *Proceedings of the Twentieth National Information Systems Security Conference*, Oct. 1997.
[15] Ulf Lindqvist and Phillip A Porras, "Detecting computer and network misuse through the production-based expert system toolset (P-BEST)," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy*, Oakland, California, may 1999, pp. 146–161, IEEE Computer Society Press, Los Alamitos, California.
[16] D. B. Lenat, "CYC: a large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, Nov. 1995.