Proceedings of the 2002 IEEE
Workshop on Information Assurance and Security
United States Military Academy, West Point, NY, 17–19 June 2002

Distributed Intrusion Detection for Policy-Controlled Heterogeneous Environments

Stephen D. Wolthusen

Abstract—**This paper describes the intrusion detection aspects of a security architecture for distributed heterogeneous systems based on a network of externalized reference monitors defining a set of policies formulated as formulae of a first order theory. This can be retrofitted onto existing operating systems or realized standalone. Aspects considered in this paper include the effects of fine-grained component-level instrumentation of the operating system and a common entity naming model imposed by the architectural framework and discusses the application of the JDL multisensor data fusion model in the context of the framework.**

Keywords—**Security Policy, Intrusion Detection, Distributed Systems**

I. Introduction

THE considerable popularity gained by intrusion detection both in research and in the field together with continued reliance on systems of dubious reliability, survivability, security, and assurance can be a cause for concern in that the perceived benefits from such an approach may preclude necessary adjustments in the defensive posture by other means. In fact, it appears that the use of COTS systems even in sensitive areas together with a defense consisting mainly of firewalling and ID systems threatens to eclipse much of the progress made earlier regarding sound models of security and verifiable implementations [1].

As has been argued before [2] the changing threat environment and an increased reliance on interdependent (frequently while unaware of the existence of such dependencies) distributed systems further calls into doubt the defensive mechanisms currently deployed particularly in the areas of operating system and application access as well as behavioral controls despite the successful efforts of research to offer mechanisms providing higher efficacy and ultimately even efficiency.

This apparent resistance to improvements in the information assurance area for general purpose systems combined with a seemingly inexorable trend towards complexity unchecked by verifiable correctness, safety, and security properties ensures a rich target selection for attackers for the foreseeable future [3].

Despite this analysis and corresponding negative outlook, risks to such COTS-based information systems need to be mitigated. In an attempt to permit a pragmatic approach to providing information assurance mechanisms even in situations where replacement of current systems by high assurance components are not feasible, the security architecture described in [2], [4], [5], [6] can be implemented in such a way that its enforcement mechanisms are retrofitted onto existing COTS systems, permitting operation in a heterogeneous environment.

S. Wolthusen is with the Security Technology Department, Fraunhofer-IGD, Darmstadt, Germany. E-mail: wolt@igd.fhg.de.

Such retrofitted systems can operate with existing application programs and using a familiar environment for end users, thereby reducing the impediments confronted by mechanisms requiring larger scale modifications and possibly paving the way for high assurance systems in the process.

This paper discusses the implications of such a distributed security policy environment for intrusion detection in several areas. Section II outlines the core concepts of the security architecture as required for subsequent discussion while sections III and IV cover some implications of the architecture for intrusion detection. Section V describes the contributions to a multisensor data fusion scenario. Finally, section VI discusses operational models for the use of policy mechanisms in enforcement and intrusion detection, particularly as reactive componentes.

II. Architectural Framework

In the framework discussed here, there exist a number of nodes called external reference monitors (ERM) which are repositories for one or more security policies, each presumably derived from a security model. The system on which an ERM resides is called a Policy Controller Node (PCN). The other component of the framework consists of a number of nodes which are subject to the policies of one or more ERM.

The policies obtained from ERM are enforced through externally controlled reference monitors (ECRM) and its enforcement modules (EM); a system configured with a combination of ECRM and EMs is called a Policy Enforcing Node (PEN). As implied by the term reference monitor, each operation of the controlled nodes is mediated by the ECRM and may only proceed if it is found to be in compliance with all applicable policies.

Applicable policies (and hence the ERMs to be consulted) are determined from the identity of subjects and objects involved which are uniquely identified by the conjunction of a subject identity and a subject type constant. Both constants are embedded in a lattice; an ERM is authorized to issue a policy if and only if it dominates the objects involved in an operation and ERM nodes on vertices dominating the node's vertex have not imposed additional constraints.

Policies are formulated in a formal theory, namely a first order predicate calculus. The subjects and objects are, as noted above, represented by constants which are grouped into sets by predicates establishing equivalence relations. Behavior permitted by a policy is then expressed as additional functions, constants, and predicates.

By identifying relations on subjects, objects, and operations and formulating these in the form of predicates one can not only express arbitrary security policies and models within a common, consistent scheme but also use automated deduction mechanisms to derive additional statements and instances. This permits the formulation of a pure security model and the relations and having the deduction mechanism derive all logically valid instances as well as formulating security policies and models in a sufficiently abstract manner such that the complexity of the specifications to be created by human security officers is limited.

It should be noted that the elements of the formal theory do not, besides the use of suggestive names, carry semantic meaning. Rather, for each instance where the mechanism is to be applied (i.e. operating systems in the case of this discussion) there must be an interpretation which supplies the semantics. If one is able to formulate the security models and policies in the form of such abstract statements, an obvious direct consequence of this approach is that the semantics of the security policy enforced are the same regardless of the specific operating system used.

To permit the placement of an upper boundary on the communication complexity imposed by the ERM/ECRM mechanism, each reply to a policy request must contain a lifetime λ specified as an ordered pair of time values. An ECRM or a caching ERM may use a reply without additional policy requests to the issuing ERM for the duration of the lifetime.

A special case exists in the case where both elements of λ are equal; this implies that the reply may not be cached at all; another exception to the lifetime is the renewal of a policy for a given ERM which revokes the previous policy and all derived rules.

When considering only the operation as a reference monitor, the behavior of the ECRM is reactive. An EM may detect a mediated operation and refer this operation to the ECRM which reformulates the operation in the form of a hypothesis. The ECRM must then consult its locally cached policy information (if any) by attempting to derive the hypothesis from the existing set of policy statements. When successful, the request is granted and may proceed. If no local policy is applicable or sufficient, the ECRM must then pose the hypotheses to all applicable ERM. An ERM may reply with a negative reply tuple (indicating that the hypothesis cannot be deduced from the policy) or a nonempty set of reply tuples which may then in turn be cached by the ECRM for the lifetime of each reply tuple since the replies issued by an ERM may contain the hypothesis posed and be usable in a large number of other mediated requests in addition to the original hypothesis.

This architecture permits the description of all machine--representable security policies and policies related to intrusion detection since a first order predicate calculus can be used to model a Turing machine. While this also implies that, given an arbitrary set S of first order formulae and a and a first order formula (hypothesis) H, it is not possible to determine whether or not $S \models H$ since this would be equivalent to solving Hilbert's *Entscheidungsproblem*.

However, this should not be seen as a drawback since safety properties are not violated and Harrison, Ruzzo, and Ullman have shown that it is undecidable whether a given state of a given protection system is secure for a given generic right [7]. For a detailed description of the ERM/ECRM mechanism see [4].

III. Effects of Component-Level Instrumentation

Within the framework just discussed each node in a distributed system can be described as a set of components or modules interacting with each other and external nodes regardless of whether the node is a general purpose computer system or fulfills only specific tasks such as a router. In the following the discussion is limited to general purpose systems without loss of generality.

Fine granularity observations of system behavior have been shown to yield results particularly valuable for anomaly detection even without the use of computationally complex processing [8]. The policy enforcement requires mediation of all operations which can be performed on entities of the policy; we assume that the same subjects, objects, and operations are also the ones relevant to auditing and intrusion detection since otherwise a contradiction would be obtained. The obvious drawback of such instrumentation is the volume of observations that must be handled; naïve centralized collection of all unprocessed observations from network nodes is not feasible in real time and impractical even for forensic purposes.

As a direct consequence processing at the source of the observations is required; this occurs under the control of the ECRM which is also the controlling entity for ID-related activities. This processing can consist of several elements which may also be used in conjunction:

1. The ECRM can modulate sensor output and level of detail based on existing and deduced policy rules; this is obviously more efficient than filtering sensor data post facto. While a certain performance loss is incurred due to the presence of the instrumentation point and determination of sensor activation, this penalty is significantly lower than postprocessing.
Since the processing steps are controlled by the ECRM (or ERMs, respectively) the modulation can occur dynamically as required.
2. The collection of observations or arbitrary policy decisions based on such observations can induce the collection of derivative information from the same or other modules and components required for additional processing and decisions.
3. Based on active policies, one or more ERMs can be contacted. The information transmitted may include observation data as well as preprocessed data and derived observations. The method for transmission of such information is also subject to policy control and may stipulate transmission of some data with minimum delay while other data can be collected at the node and transmitted at a later point in time in a burst together with other data.

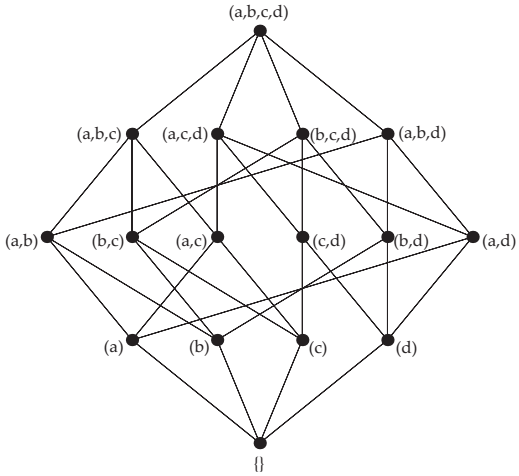

Fig. 1. Hasse diagram for the lattice $\mathcal{P}(\{\mathrm{a},\mathrm{b},\mathrm{c},\mathrm{d}\})$ ordered by set inclusion

IV. Effects of Common Entity Naming Model

The foundation for the security policy consists of a set of predicates defining abstract properties. Interpretations establish equivalence relations over native primitives of the systems to which the policy is to apply. The predicates themselves map onto the type lattices.

As noted in section II, all constants identifying subjects and objects correspond to elements of two lattices (a partially ordered set in which every pair of elements has a least upper bound and a greatest lower bound); this can be visualized in a Hasse diagram (figure 1 shows the diagram for the lattice $\mathcal{P}(\{\mathrm{a},\mathrm{b},\mathrm{c},\mathrm{d}\})$ ordered by set inclusion — the exact choice of lattice may be different depending on the model chosen or not all vertices need to be present) where each entity is mapped to a vertex in the diagram.

While a single lattice (for identity) is sufficient, the use of a second lattice permits a more concise representation of which the former can be considered an extension expressing the equivalence class.

The type lattice permits a compact representation of predicates, functions, and constants formulating the security policy since one can express both that an entity is derived from another (if an x is contained in y) and thus may be considered semantically subordinate. An example of a member of this relation would be expressed by the equivalence classes represented by the predicates $\mathrm{datagram}(\mathrm{x})$ and $\mathrm{virtual\_circuit}(\mathrm{y})$.

The same principle also applies to the representation of subordinate entities of ephemeral nature; the ability to tie the identity of an entity in the $\mathrm{process}(\mathrm{x})$ equivalence class to a specific $\mathrm{user}(\mathrm{y})$ permits the efficient reconstruction of such relations without what would otherwise amount to a simulation of a system's semantics.

Similarly, since entity identities are represented as elements of a lattice (i.e. vertices of the Hasse diagram), this permits the constructions of globally unique identifiers within the distributed system while retaining the ability to group entities according to the least upper bounds within the specific lattice structure (another example for a lattice would be a tree graph, appropriate for hierarchical structures).

The first immediate result of this is the ability to express security policies in terms of the equivalence class predicates regardless of the underlying interpretation (i.e. host operating system or application) provided such an interpretation exists as was noted above.

As a corollary to this, observations from nodes can now also be represented in the form of such equivalence classes. While of no particular relevance in the case of observations from an individual node where all observations are within the domain of a single interpretation, this gains importance once observations from multiple nodes with heterogeneous interpretations must be taken into consideration.

Unless there is a common model for observations (at least at a certain abstraction level, see section V) within a heterogeneous system, the individual observations are incommensurate and contribute to the overall complexity of an intrusion model.

A. Enforcement Module Layering

Despite widely divergent appearances, most deployed operating systems share fundamental abstractions and mechanisms which have remained essentially stable for several decades. This permits the use of the previously described naming schema for entities even across system boundaries.

Similarly, the mechanisms for representing basic abstractions such as files and application programs are sufficiently similar to permit the modeling of such abstractions in terms of layered enforcement modules such as for network interface, network protocol stack, and file system (and hence also instrumentation points).

While, as discussed in [2], the primary purpose for this layering is the reduction in complexity for policy specification, a desirable side effect is the ability to correlate observations particularly at lower abstraction levels with instrumentation data from other layers. This permits the reduction of isolated observation data and provides characteristic patterns of instrumentation point activation that are not available by instrumentation at the system call level alone.

V. The Data Fusion Model

Data fusion [9] as defined in [10] is "the process of combining data or information to estimate or predict entity states". The model depicted in figure 2 consists of several levels, here slightly revised from the model presented in [9]:

Level 1: Object Refinement Aimed at combining sensor data to obtain reliable and accurate estimates of an entity's identity and properties.

Level 2: Situation Refinement Dynamically attempts to develop a description of current relationships among entities and events in the context of their environment.

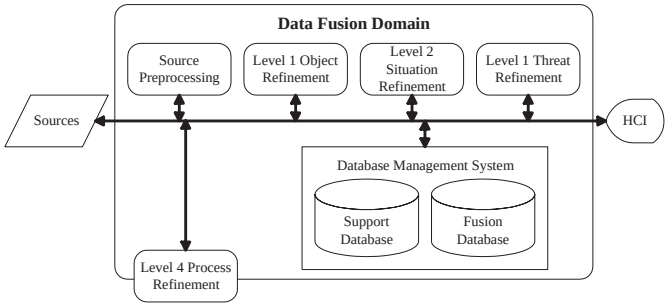

Fig. 2. The Joint Directors of Laboratories (JDL) process model for data fusion

Level 3: Threat Refinement Projects the current situation into the future to draw inferences about enemy threats, friend and foe vulnerabilities, and opportunities for operations.
Level 4: Process Refinement A meta-process that monitors the overall data fusion process to assess and improve real-time performance.

While primarily developed for real-time defense applications involving physical targets (which have some properties not applicable to IDS targets such as a kinematics envelope permitting to rule out certain sensor data), the paradigm clearly has merit for modeling advanced intrusion detection mechanisms [11].

Most current IDS operate on raw data material such as network traffic or operating system audit trails; typically the latter is not specific to intrusion detection but rather intended to meet auditing requirements such as the CAPP [12], [13] (formerly TCSEC C2 [14]) and does not match the requirements of the IDS in volume and granularity. This also counteracts the approach of using C2 auditing as the baseline for unifying heterogeneous systems as apparently became evident during the development of DIDS [15].

A. Level 0

Source Preprocessing (also referred to as Level 0) in this model occurs at the level of individual PENs. and must in itself be separated into three processing sublevels.

At the lower sublevel (0A), each enforcement module is equipped with a number of instrumentation points, i.e. locations in code where a number of items can be collected. These include the fact that the instrumentation point was reached in the flow of execution as well as information available at the particular point such as the process and thread IDs as well as the account under which the executing process is running. These instrumentation points represent the source for raw data, no homogenization occurs at this stage.

Each of the instrumentation point is uniquely identified and can therefore be activated as necessary. In cases where the auxiliary data collected must be configurable (e.g. in the case of collecting network traffic), this must be modeled by several separate instrumentation points.

To reflect this as well as to permit the creation of classes of sensor activation, instrumentation points can be coalesced into instrumentation groups; the sets thus formed may intersect.

At the middle sublevel (0B) the data collected by the instrumentation points is processed by annotating the lower sublevel data with the type and identity constants wherever these are known. The rationale for the insertion of this sublevel is that, while some audit or intrusion detection mechanisms and policies may require the use of raw instrumentation data, significant parts of these contain ephemeral references to subjects and objects or are not unique.

At the upper sublevel (0C) the data from one or more instrumentation points is translated and collated into predicates of the formal theory defining operations in the terms of security policies. The type of translation required depends on the system on which it occurs. In some cases a bijective mapping between predicates and specific system calls or operations exists; in others a sequence of calls or operations must fulfill certain criteria such as the presence of certain parameters in all elements of the sequence. This requires that the ECRM retains a history of operations and the entities involved in the operations from which a sequence of operations can be determined which can e.g. be done efficiently using colored Petri nets (CP-nets) [16], [17], also permitting the introduction of temporal constraints on the matching. The semantics of CP-nets permit the representation of fully concurrent operations and offer a well understood mechanism for confluent preconditions that can be modeled and the derivation of CP-nets from well-formed formulae in formal can be done automatically and verified rigorously.

The relative cost of retaining the operation history is further reduced by the fact that the inverse to the mapping which must be performed for the Level 0 ID processing is required for security policy enforcement, the overhead for retaining this information for intrusion detection is therefore negligible.

Unless annotated level 0B data is used, this MDF layer provides a data reduction mechanism and a normalized set of features independent of the host operating system of the sensor platform.

Intrusion detection mechanisms (particularly anomaly detection systems) may find the preprocessed data sufficient and can still benefit from the consequences outlined in section IV as well as from the ability to dynamically modulate sensor output.

B. Level 1

The goal of determining the identity of an entity, particularly of subjects, associated with level 1 data fusion in the JDL model is already achieved by the normalization in level 0B for entities falling under the direct control of the policy enforcement mechanisms. Entities outside this domain of control cannot be implicitly identified positively.

For such entities it becomes necessary to perform level 1 data fusion if events related to more than one activity are to be correlated. This level establishes hypotheses based on observations normalized by level 0 processing as well as existing hypotheses that a certain set of observations represents an distinct individual entity; this is also referred to as a “track”.

Such a track may consist of a cluster of operations centered around a certain object or set of objects as established by an anomaly detection mechanism from which a heuristic may determine that a common factor (i.e. the hypothesized subject) is the cause for the clustering occurrence.

Another technique that can be used for level 1 fusion may be termed an inverse intrusion signature mechanism. While one is typically interested in the fact that an intrusion may have occurred, the general mechanism of identifying characteristic operations can also be used in such a way that the pattern of observations give rise to the hypothesis that this signature is associated with a single causative entity.

A major difficulty for level 1 fusion is the ease with which masquerading can occur, both inadvertently or deliberately as a decoy. One example for such masquerading is the use of readily available network scanning tools which will generally use address spoofing or use intermediate compromised hosts to mount reconnaissance operations.

The pattern of observations will then contain a large amount of correlated features although there may be several independent subjects involved. Given the large number of such reconnaissance operations and subjects, this appears present a particularly urgent need for taking additional characteristics from different instrumentation points or sensors into account in firming up the identity hypothesis for a subject. In the example mentioned here such characteristics would include temporal patterns such as apparent network latency.

The relevance of determining the identity of a subject presumed to be involved in hostile actions is twofold. One is in situations where offensive information operations against hostile subjects may be both justified and feasible; here the primary interest is in ascertaining the identity with the least degree of uncertainty possible. It should be noted that this is only feasible if the enforcement portion of the security architecture is employed in such a way as to permit positive identification and authentication of remote entities (primarily using cryptographic techniques[2]) due to the potential for mistakenly responding to e.g. forged network addresses. This permits the a priori classification of a large portion of entities into FFN categories. Entities which have been identified positively cannot, however, be statically classified as friendly or even neutral due to the possibility of these entities being subverted.

The remaining necessity arises when one wishes to perform higher level (2 and 3) data fusion; here the entity identification is a necessary prerequisite.

Level 1 fusion is feasible relying on the sensor suite of an individual node; while in some cases information stemming from other nodes may be desirable, the benefits of such cross-node fusion at level 1 is as yet conjectural and would constitute a significant increase in complexity.

C. Level 2

During level 2 fusion a number of tracks or hypothetical entities are aggregated into what is commonly referred to as a situation.

The representation of such situations can be accomplished using the primitives established earlier in that relations are modeled in the first order theory. These can be:

- Aggregation relations which are modeled as predicates establishing equivalence classes over entities.
- Temporal relations which cannot be expressed naturally in the context of a first order predicate logic but either using predicates modeling temporal relations (but not permitting inference on these; this becomes relevant in level 3 fusion) or using temporal logic. The latter is rather unproblematic since temporal or indeed most modal logic systems — with some exceptions such as propositional dynamic logic that are presumably of limited interest in this context — can be efficiently translated into first order logic [18], [19], [20].
- Causal relations which are expressed as connectives in the formal theory that need not be constrained to implication and equivalence.
- Similarity relations may be considered a special case of aggregation relations which are qualified additionally by a similarity metric that in turn is dependent on the identity or the equivalence class membership of the entities involved.

In most cases a probability metric is required for the relations; while this metric can be modeled as a partial function over the elements of the relations, such an approach is clearly unappealing.

Instead one can follow [21] and obtain a mathematically sound method for inferring relational probability metrics, namely the use of a Bayesian network for obtaining a distribution of discrete states x_d for an entity state X under the assignment to the node in a level 2 hypothesis ζ combining estimated entity states X_i under level 1 hypotheses based on observations Z_i and estimated entity states X_i in a level 2 hypothesis based on a set of relations R_i among tracks. It should be noted, though, that inference is not restricted to elements on the same fusion level but may also include lower level hypotheses.

Somewhat problematic in this is the proper assignment of probabilities to individual hypotheses which can aggregate into significant uncertainty given the high dimensionality of the state space. We assume that this significantly curtails the efficacy of inferences that can be drawn automatically compared to fusion scenarios based on physical observations.

Alternatives, particularly in the field of logic involve the use of non-monotonic logic [22], [23], [24]; this also permits the use of epistemic logic under the same framework. Even when dealing with decidable aspects of non-monotonic logics as is typically the case in knowledge representation the problems of highly nondeterministic reasoning and complexity remain [25].

Specification of knowledge producing actions such as observations and belief revision within the framework of multi-modal logic has been the subject of research [26], [18] as has been the case for fuzzy logic [27], [28].

D. Level 3

Level 3 fusion is mainly concerned with predicting the state space established by the hypotheses at level 2 at a point in time in the future and is intended for the estimation of impact of courses of action such as the likelihood of an outcome state and cost metrics for such actions.

The primary area of interest here is presumably the interactive evaluation of scenarios; given the high degree of uncertainty introduced at level 2 it is not immediately apparent how such a mechanism could e.g. be employed for automated reactive behavior without incurring significant risks of aberrant behavior.

VI. Operational Models

A number of scenarios are possible when combining the aspects of policy-controlled enforcement and intrusion detection; these are outlined below. In the following discussion, all scenarios include mechanisms introduced by previous scenarios:

Passive Detection This situation is equivalent to the deployment scenario generally used for IDS, namely intrusion detection decoupled from policy enforcement. The node is fully exposed to all threats from undesirable behavior of insiders as well as to external attacks.

Here the benefits of having extensive instrumentation at the various abstraction layers can be assumed to be roughly equivalent to that provided by [8] and [29] in that they provide a more detailed view of host-based system behavior than what can be expected from native instrumentation typically intended for TCSEC C2 or later equivalents.

For network-based components, the immediate benefit is in the ability to access network PDUs that are subject to end-to-end encapsulation or encryption and in distributing the processing load, addressing bandwidth limitations at central detectors in the process.

In this scenario a policy can be distributed ad hoc and ensuring equivalent behavior across heterogeneous platforms including possible preprocessing before reporting to designated processing and fusion nodes.

Augmented Detection In this scenario policy enforcement is also decoupled from intrusion detection. Hence, the threat environment is as described in the previous scenario.

However, a dynamic element is introduced at two levels. First, local policy rules can, in response to observations from local sensors, derive additional rules for activation of sensors. This implicitly includes the derivation of intrusion scenarios using the deduction system (which amounts to the use of the deduction engine as a production system) or a secondary anomaly detector capable of generating rules as output. All such rules are implicitly time-bounded using the lifetime mechanism described earlier. The second dynamic element consists of the ECRM nodes sending sensor data or derived hypotheses to ERM nodes which then react to these notifications from one or more nodes. Again, the reaction on the part of the ERM can be induced by the deduction system or an anomaly detector. Policy elements for intrusion detection such as an increase in output volume for certain sensors on specific nodes can thus be obtained and propagated to the ECRM nodes.

It should be noted that the extent to which the sensor activation, output modulation, and subsequent processing by both ERM and ECRM takes place must be weighed carefully since degradation or denial of service can otherwise occur.

Enforcement Augmentation In this scenario, security policy enforcement is in place and is merely augmented by intrusion detection capabilities; this represents the intended environment for the architecture described in this paper. In the presence of policy enforcement, some premises for the previous scenario are — depending on the policies — no longer applicable.

Policies which enforce that only legitimate operations are performed under a given security model and clearly sets of operations and the circumstances under which these may be performed can a priori eliminate a significant amount of behavior that would otherwise need to be analyzed for signs of intrusion as well as behavior which, while legitimate under the security policy, is abnormal.

Some of this ambiguous area for a given set of policies can probably be covered by deduction-based detection; most, however, will need to be analyzed using anomaly detection. Attempts at violating the security policies need to be analyzed in conjunction with legitimate behavior preceding it or concomitant to it; it is mainly in the elimination of some behavior subsequent to initial breaches of policy that would otherwise need to be analyzed forensically.

Attack detection (i.e. operations performed by unauthorized subjects not under the control of policy mechanisms) represent a category which still needs to be addressed. However, given the volume of sensor data and particularly of observations which require further processing, the distributed processing or preprocessing becomes highly valuable even though the amount of sensor data is otherwise reduced.

Fully Reactive While the drawbacks inherent in this scenario are severe and will presumably preclude it from consideration in most cases, it is included here nonetheless for completeness. This scenario adds definitions to the policy rule sets for rewriting policies to restrict operations normally within the purview of certain subjects or concerning sensitive objects (e.g. limiting access to objects with a given classification on a node for which attacks followed by anomalous behavior of internal subjects has been observed).

Even more so than in the Augmented Detection and Enforcement Augmentation scenarios with reactive sensor and processing behavior this creates severe risks of denial of service for legitimate subjects.

Finally, it is also conceivable that the Fully Reactive scenario is extended through the policy-based activation of active information operations; since this would only be relevant in situations where external attacks are detected this implies that in all likelihood the identity of the adversary cannot be positively verified. This, however, almost ensures that a skilled adversary will be able to use predictable reactive systems against the defender.

VII. Related Work

Research on distributed intrusion detection has been ongoing for more than a decade; one of the earliest, DIDS [15], [30], was developed at the University of California at Davis and combined distributed monitoring and data reduction (through individual host and LAN monitors) with centralized data analysis (through the DIDS director) to monitor a heterogeneous network of computers. This was based on the assumption that systems intended for TCSEC C2 evaluation would have similar audit trails that could be used as observations.

NIDES and EMERALD [31], [32] both developed at SRI following seminal work by Neumann and Denning on IDES [33] also pursue the model of running monitor agents to which NIDES added a translation layer on the monitored node for conversion of audit records into the NIDES native format; this approach has since been used by a number of other systems as well.

EMERALD represents a framework for multiple intrusion detection components which strives to separate the analysis from the collection of observations. Hence it is conceivable that the instrumentation mechanism (albeit limited to operation without feedback to the PENs) provided by the architecture described in this paper can act as an EMERALD monitor.

The first order logic automated deduction mechanism can be used analogous to a production system whose use was pioneered in MIDAS and IDES [33], [34] albeit with a larger expressiveness.

DPEM, also developed at UC Davis is presumably the first instance of a specification-based system based on earlier work on execution monitoring [35], [36] intended for privileged Unix programs. It monitors several aspects of program behavior such as access to system objects, sequencing, synchronization, and race conditions. Related experience in this area is also discussed in [37]. It should be noted that this approach is in some ways the inverse of what is discussed in this paper in that the specification is not enforced but used only for detecting misuse. Fraser et al. pursued a related approach [38] for specification-based security enforcement.

VIII. Conclusions and Future Work

We have discussed an architectural framework for both security policy enforcement and intrusion detection instrumentation and control that is also suitable for retrofitting to existing systems.

Following the argumentation in [11], the suitability of the mechanisms for a multisensor data fusion suite was discussed; we particularly hope that the architectural framework will permit the development of level 2 data fusion eventually leading to situational awareness within distributed systems — within the confines of the uncertainty imposed by the high dimensionality and lack of constraints encountered in other fusion scenarios.

Ongoing research concentrates on providing highly instrumented enforcement modules for a number of components for COTS operating systems, particularly the commercially relevant Microsoft Windows NT and Unix System V Release 4 operating system families as well as on efficient automated deduction systems suitable for integration into standalone, verifiable systems.

References

[1] Roger R. Schell, "Information Security: Science, Pseudoscience, and Flying Pigs," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)*, New Orleans, LA, USA, Dec. 2001, pp. 205–216, IEEE Computer Society Press.

[2] Stephen Wolthusen, "Layered multipoint network defense and security policy enforcement," in *Proceedings from the Second Annual IEEE SMC Information Assurance Workshop, United States Military Academy*, West Point, NY, USA, June 2001, pp. 100–108, IEEE Press.

[3] Robert M. Brady, Ross J. Anderson, and Robin C. Ball, "Murphy s law, the fitness of evolving species, and the limits of software reliability," Tech. Rep. 476, Cambridge University, Cambridge, UK, Computer Laboratory, 1999.

[4] Stephen Wolthusen, "Access and Use Control using Externally Controlled Reference Monitors," *ACM Operating Systems Review*, vol. 36, no. 1, pp. 58–69, 2002.

[5] Stephen Wolthusen, "Security Policy Enforcement at the File System Level in the Windows NT Operating System Family," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)*, New Orleans, LA, USA, Dec. 2001, pp. 55–63, IEEE Computer Society Press.

[6] Stephen Wolthusen, "Embedding Policy-Controlled ID Sensors within Host Operating System Security Enforcement Components for Real Time Monitoring," in *Proceedings of the NATO RTO Symposium on Real-time Intrusion Detection Symposium (RTID)*, Lisbon, Portugal, May 2002, NATO Research and Technology Organization, To appear.

[7] Michael A. Harrison, Walter L. Ruzzo, and Jeffrey D. Ullman, "Protection in Operating Systems," *Communications of the Association of Computing Machinery*, vol. 19, no. 8, pp. 461–471, Aug. 1976.

[8] Christina Warrender, Stephanie Forrest, and Barak Pearlmutter, "Detecting Intrusions Using System Calls: Alternative Data Models," in *Proceedings of the 1996 IEEE Symposium on Security and Privacy (SOSP '96)*, Oakland, CA, USA, May 1996, pp. 133–145, IEEE Computer Society Press.

[9] Franklin E. White, Jr., *Data Fusion Lexicon*, Joint Directors of Laboratories, Technical Panel for C^3, Data Fusion Subpanel, Naval Ocean Systems Center, San Diego, CA, USA, 1987.

[10] Alan N. Steinberg, C. L. Bowman, and Franklin E. White, Jr., "Revisions to the JDL Data Fusion Model," in *Proceedings of the 3rd NATO/IRIS Conference*, Quebec City, Canada, Oct. 1998, NATO Research and Technology Organization, Proceedings are classified, summarized in [39].

[11] Tim Bass, "Intrusion Detection Systems and Multisensor Data Fusion," *Communications of the ACM*, vol. 43, no. 4, pp. 99–105, Apr. 2000.

[12] ISO/IEC Standard 15408, *Common Criteria for Information Technology Security Evaluation*, Dec. 1999, Version 2.1.

[13] National Security Agency Information Systems Security Organization, *Controlled Access Protection Profile*, Fort George G. Meade, MD, USA, Oct. 1999, Version 1.D.

[14] United States Department of Defense, *DoD 5200.28-STD: Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC)*, 1985.

[15] S. R. Snapp, J. Brentano, G. V. Dias, T. L. Goan, L. T. Heberlein, C.-L. Ho, K. N. Levitt, B. Mukherjee, S. E. Smaha, T. Grance, D. M. Teal, and D. Mansur, "DIDS (Distributed Intrusion Detection System) – Motivation, Architecture, and an Early Prototype," in *Proceedings 14th NIST-NCSC National Computer Security Conference*, Washington D.C., USA, 1991, pp. 167–176.

[16] Kurt Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use, Volume 1*, Monographs in Theoretical Computer Science. Springer-Verlag, Heidelberg, Germany, 1997.

[17] Kurt Jensen, *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use, Volume 2*, Monographs in Theoretical Computer Science. Springer-Verlag, Heidelberg, Germany, 1997.

[18] Hans Jürgen Ohlbach, "Optimized Translation of Multi-Modal Logic into Predicate Logic," in *Proceedings of the 4th Conference in Logic Programming and Automated Reasoning (LPAR '93)*, Andrei Voronkov, Ed., St. Petersburg, Russia, July 1993, vol. 698 of *Lecture Notes in Computer Science*, pp. 253–264, Springer Verlag.

[19] Renate A. Schmidt, "Decidability by Resolution for Propositional Modal Logics," *Journal of Automated Reasoning*, vol. 22, pp. 379–396, 1999.

[20] Ullrich Hustadt and Renate A. Schmidt, "MSPASS: Modal Reasoning by Translation and First-Order Resolution," in *Automated Reasoning with Analytic Tableaux and Related Methods — International Conference TABLEAUX 2000*, R. Dyckhoff, Ed., St. Andrews, UK, July 2000, vol. 1847 of *Lecture Notes in Computer Science*, pp. 67–71, Springer Verlag.

[21] Alan N. Steinberg and Robert B. Washburn, "Multi-level fusion for Warbreaker intelligence correlation," in *Proceedings of the 8th National Symposium on Sensor Fusion*, 1995.

[22] Peter Gärdenfors, "Belief Revision and Nonmonotonic Logic: Two Sides of the Same Coin ?," in *Proceedings of the European Workshop on Logics in AI (JELIA'90)*, J. van Eijck, Ed., Amsterdam, the Netherlands, Sept. 1990, vol. 478 of *Lecture Notes in Computer Science*, pp. 52–54, Springer Verlag.

[23] Bernhard Nebel, "Syntax-Based Approaches to Belief Revision," in *Belief Revision*, P. Gärdenfors, Ed., vol. 29 of *Cambridge Tracts in Theoretical Computer Science*, pp. 52–88. Cambridge University Press, Cambridge, UK, 1992.

[24] Didier Dubois and Henri Prade, Eds., *Handbook of Defeasible Reasoning and Uncertainty Management Systems: Belief Change*, vol. 3, Kluwer Academic Publishers, Dordrecht, the Netherlands, 1998.

[25] Marco Cadoli and Marco Schaerf, "A Survey of Complexity Results for Non-Monotonic Logics," *Journal of Logic Programming*, vol. 17, pp. 127–160, 1993.

[26] Bernd van Linder, Wiebe van der Hoek, and John-Jules Meyer, "Actions that Make you Change Your Mind," in *Knowledge and Belief in Philosophy and Artificial Intelligence*, Armin Laux and Heinrich Wansing, Eds., pp. 103–146. Akademie Verlag, Berlin, Germany, 1995.

[27] Lotfi A. Zadeh and Janusz Kacprzyk, Eds., *Fuzzy Logic for the Management of Uncertainty*, John Wiley & Sons, New York, NY, USA, 1992.

[28] Susan M. Bridges Jianxiong Luo, "Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection," *International Journal of Intelligent Systems*, vol. 15, no. 8, pp. 687–703, 2000.

[29] John C. Munson and Scott Wimer, "Watcher: The Missing Piece of the Security Puzzle," in *Proceedings 17th Annual Computer Security Applications Conference (ACSAC'01)*, New Orleans, LA, USA, Dec. 2001, pp. 230–239, IEEE Computer Society Press.

[30] Steven R. Snapp, Stephen E. Smaha, and Daniel M. Teal, "The DIDS (Distributed Intrusion Detection System) Prototype," in *Proceedings of the Summer 1992 USENIX Conference*, USENIX Association, Ed., San Antonio, TX, USA, June 1992, pp. 227–234, USENIX.

[31] Raj Jagannathan, Teresa Lunt, Debra Anderson, Chris Dodd, Fred Gilham, Caveh Jalali, Hal Javitz, Peter G. Neumann, Ann Tamaru, and Alfonso Valdes, "System Design Document: Next-Generation Intrusion Detection Expert System (NIDES)," Tech. Rep. A007/A008/A009/A011/A012/A014, SRI International, Menlo Park, CA, USA, Mar. 1993.

[32] Phillip A. Porras and Peter G. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances," in *Proceedings 20th NIST-NCSC National Information Systems Security Conference*, Baltimore, MD, USA, Oct. 1997, pp. 353–365.

[33] Dorothy E. Denning and Peter G. Neumann, "Requirements and Model for IDES – A Real-Time Intrusion Detection Expert System," Tech. Rep., Computer Science Laboratory, SRI International, Menlo Park, CA, 1985.

[34] Michael M Sebring, Eric Shellhouse, Mary E. Hanna, and R. Alan Whitehurst, "Expert System in Intrusion Detection: A Case Study," in *Proceedings of the 11th National Computer Security Conference*, Baltimore, MD, USA, Oct. 1988, pp. 74–81.

[35] Calvin Ko, George Fink, and Karl Levitt, "Automated detection of vulnerabilities in privileged programs by execution monitoring," in *Proceedings of the 10th Annual Computer Security Applications Conference (ACSAC '94)*, Orlando, FL, USA, Dec. 1994, pp. 134–144, IEEE Computer Society Press.

[36] Calvin Ko, Manfred Ruschitzka, and Karl Levitt, "Execution Monitoring of Security-Critical Programs in a Distributed Systems: A Specification-Based Approach," in *Proceedings of the 1997 IEEE Symposium on Security and Privacy (SOSP '97)*, Oakland, CA, USA, May 1997, pp. 175–187, IEEE Computer Society Press.

[37] Prem Uppuluri and R. Sekar, "Experiences with Specification-Based Intrusion Detection," in *Recent Advances in Intrusion Detection: 4th International Symposium (RAID 2001) Proceedings*, W. Lee, L. Mé, and A. Wespi, Eds., Davis, CA, USA, Oct. 2001, vol. 2212 of *Lecture Notes in Computer Science*, pp. 172–189, Springer Verlag.

[38] Timothy Fraser, Lee Badger, and Mark Feldman, "Hardening COTS Software with Generic Software Wrappers," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy (SOSP '99)*, Oakland, CA, USA, May 1999, pp. 2–16, IEEE Computer Society Press.

[39] David L. Hall and James Llinas, Eds., *Handbook of Multisensor Data Fusion*, The Electrical Engineering and Applied Signal Processing Series. CRC Press, Boca Raton, FL, USA, 2001.