

Using data correlation to build an intrusion detection system

L.ROMANO*[?], V.VIANELLO*^α, S.D'ANTONIO*^α, S.GIORDANO*^α

* Dipartimento per le Tecnologie
Università degli Studi di Napoli "Parthenope"
Centro Direzionale di Napoli – 80143 Napoli
ITALY

(lrom, valerio.vianello, salvatore.dantonio, savio.giordano)@uniparthenope.it

^α Laboratorio ITeM "Carlo Savy"
Consorzio Interuniversitario Nazionale per l'Informatica (CINI)
Via Cinthia - Edificio 1, 80126 Napoli
ITALY

[?] Istituto di Calcolo e Reti ad Alte Prestazioni
Consiglio Nazionale delle Ricerche (ICAR-CNR)
Via Pietro Castellino 111 I-80131 Napoli
ITALY

Abstract: - Intrusion Detection Systems (IDSs) are one of the most adopted technologies when facing the issue of computer security. Regrettably, current solutions are far from perfect: i) either they produce a large number of false positives or they detect only known attacks; ii) they do not scale as the monitored infrastructure grows in terms of number of components and of exchanged data. Correlation of attack symptoms from diverse information sources has been proven to be an effective approach. In this paper, we propose an IDS solution which correlates information from diverse sources for improved performance, i.e. achieving high detection while reducing false positives. We discuss the key issues that result from adopting correlation of data coming from multiple sources and present the conceptual architecture that has been drawn in the PHDS ("A Middleware Infrastructure for Real-Time Processing of Heterogeneous Data Streams") project to face such issues in a Security and Safety domain. We also present technological choices taken to implement such an architecture.

Key-Words: -"Intrusion Detection" "Log Correlation" "Information Diversity" "Stream Processing" "Grammar based parsing" "SOA"

1 Rationale

Detecting intrusions is a hard task in any networked environment, since a network naturally lends itself to distributed exploitation of its resources. In such a scenario, the identification of a potential attack requires that information is gathered from many different sources and in many different places, since no locality principle (neither spatial nor temporal) can be fruitfully applied in the most general case. For these reasons, we propose a distributed intrusion detection system (IDS) comprising a number of components whose operation is orchestrated to ensure that the network is effectively protected.

Classical approaches to distributed protection of a network rely on the effective dissemination of probes and classifiers/analyzers across the network infrastructure. We claim that the current solutions lack two fundamental features, namely diversity and correlation.

To exploit diversity it is needed to distribute a number of sensors collecting data from multiple devices into the network at different architectural levels. Data collecting is performed by using a distributed approach to network monitoring and data reduction which allows combining

events generated by different security applications (e.g. Snort IDS, log analyzer, Cisco PIX, antivirus, etc.). The high heterogeneity of both the semantic and the format of collected events pose a first challenge since it is necessary to provide suitable parsers which are able to collect events and to normalize them allowing their following processing. On the other side it is not suitable the idea of customizing parsers by hand any time a new probing point is added.

The second challenge resides in the need for correlating information coming from diverse data sources in order to improve performance and accuracy of intrusion detection systems. A correlation capability potentially improves detection performance through the aggregation of different views of the same incident provided by heterogeneous distributed sensors. On the other hand, using multiple heterogeneous sources might increase the alerts volume making it necessary an architecture able to scale as the monitored infrastructure grows in number of components and in volume of network traffic.

In this paper we present a conceptual architecture of a intrusion detection system resulting from the specialization of the PHDS middleware for the Security and Safety

domain. PHDS (“A Middleware Infrastructure for Real-Time Processing of Heterogeneous Data Streams”) is a project in which partners belonging to Academia and Industry have collaborated to design a middleware able to allow monitoring and control of complex real time processes. Besides the security and safety domain in the PHDS project has been proposed specialization of the middleware for the Telecommunication and Telecare domains too. The proposed solution is based on a detection engine which exploits Complex Event Processors (CEPs) to allow real-time correlation of huge mass of information flowing through data streams. We also presents the key technologies that have been chosen to implement such an architecture motivating our choices.

The rest of the paper is organized as follows: section 2 provides a high level view of the architecture of the proposed system; section 3 discusses the key issues that must be addressed, and provides pointers to emerging technologies in the field; section 4 provides a review of relevant research in the field; finally section 5 concludes the paper with a discussion of lessons learned and future directions of our research.

2 Conceptual architecture of a diversity-based intrusion detection system

The overall architecture of the Intrusion Detection System is depicted in Figure 1. It comprises the following components:

- one or more network-level Intrusion Detection Systems, which consist of: (i) a data broker, which is in charge of gathering information from the monitoring system and distributing the collected information to one or more detection engines, (ii) distributed detection engines, which receive data from the broker and decide on whether or not such information represents a potential attack pattern, based on a specific detection technique and (iii) a network level decision maker, which is in charge of combining information coming from multiple detection engines and analyzing diverse subsets of the source data in order to effectively detect attacks.
- a number of sensors (Adaptable Parsers), which are distributed throughout the network and feed events to the Event Bus. Events originate from multiple sources such as, web servers, firewalls, Databases, host based IDS, and so on. The number of sources entails an high level of heterogeneity in both events semantic and formats, moreover the volume of generated events rise up quickly as the number of monitored point into the system grows.
- the system level Decision Engine, which queries the Event Bus and takes the final decision on whether

or not the collected events represent a potential attack.

3 Key issues and technologies

3.1 Adaptable parsing of multiple data feeds

Nowadays information systems typically consist of distributed entities which interact according to loosely coupled cooperation schemes. While cooperating, these entities generate data streams which contain information about relevant events manifesting at several architectural levels, e.g. the network level, the Application Server level, the DBMS level, the physical level, and so on. An efficient IDS for complex enterprise environments should be capable of extracting information from the numerous online sources available.

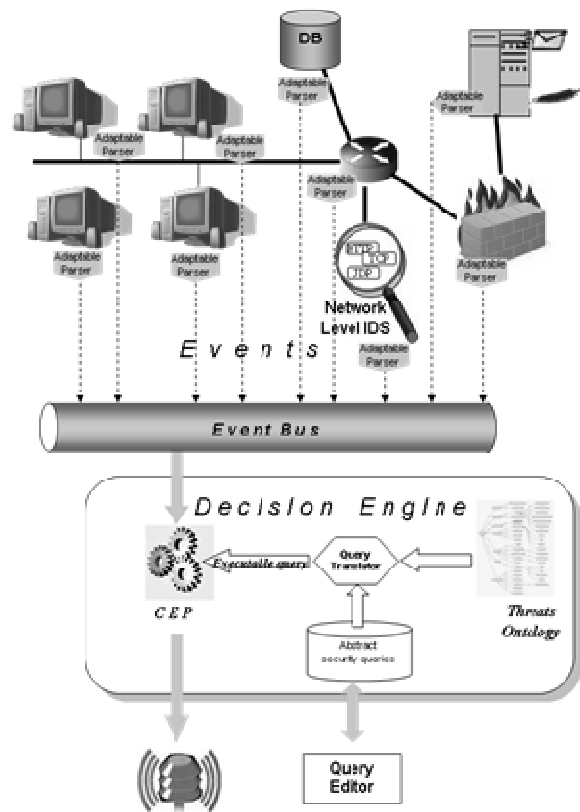


Fig. 1 - Conceptual architecture of the Intrusion Detection System

Typically, vast amounts of data are associated to these data sources and exchanged in ad-hoc formats for which off-the-shelf processing tools are not available. Normally, these information flows need to be processed as continuous streams of data. Examples of such information streams include Web server logs [1], network performance logs used in network monitoring, and other application stream data

used in specific domains, such as medical monitoring and diagnoses. Most of these data streams have an application-dependent format, typically fixed or variable column ASCII, or structured log elements expressed in such languages as XML.

Adaptable Parsers are responsible for managing the process of collecting (and converting) events which are produced by a variety of data feeds. Adaptable Parsers are based on the concept of “grammar based log analysis”, which implies

- a very large degree of expressiveness,
- the availability of well-known tools for the automatic processing of grammar-based artifacts,
- a high level of generality and technology-independence, which decouples the format definition from the underlying technology used for data processing.

The adoption of Adaptable Parsers, based on a grammar based description of events formats, provides:

- Transparency. the application logic remains unchanged. Adding a new stream and related format only entail providing an external plug-in component.
- Technology-independence. The process of extending the application with a new information source does not depend on the specific technology used to implement the application: formats are defined in a uniform, reusable, and technology-independent formalism.
- Automated design flows. Reliable tools for automatic generation of format-dependent components are available.
- Streaming. The streamed nature of information flows is not altered, e.g. by adding unnecessary synchronization points.

Grammars provide a suitable technology-independent formalism and also a set of associated off-the-shelf tools (such as Lex/Yacc, or JavaCC) for automatic generation of processing components. Grammars are totally independent of the application logic and technology. Writing a grammar only implies knowing the common semantic interface exposed by the application. Grammar definitions are used to generate “pluggable” components, one for each data format, that are then connected to the main application through an appropriate programming interface inferred from the common semantics. The multiple streams that enter the application after the conversion can then be uniformly processed, for example they can be aggregated into a single stream, and/or analyzed for extracting common global events.

As for the PHDS project, we chose the JavaCC Grammar as the formalism for data description and JavaCC [4] as the Compiler Compiler needed to obtain custom parsers. The choice is motivated by the powerful support for error handling as there is no guarantee that the rules of the grammar are strictly observed in the input stream, due to a

number of reasons, especially to “noisy” input channels [3]. JavaCC provides the typical error handling support offered by Java (i.e. Exception handling, Assertion). Moreover, JavaCC simplifies the writing of robust grammars since it supports the concept of Lexical States. Further details on Adaptable Parsers are available in [5].

3.2 Stream processing and Complex Event Processing

Effective Intrusion Detection requires the ability to analyze massive amounts of data in real-time, in order to discover facts of interest. Currently data mining platforms exhibit limited scalability both in terms of the storage capacity as well as transfer and computation throughput (despite recent developments in technology of commodity storage systems, cluster interconnects, and communication middleware, provide the potential to dramatically improve the current-state-of-the-art in such platforms). The use of (i) low-latency, high-bandwidth commodity interconnects, (ii) low-end, cost-effective disk drives attached to the network, and (iii) novel communication protocols stacks and higher-level middleware are key technologies that can be explored.

In summary, the problem prohibits to use either pure in-memory streaming engines or offline processing of network data because the amount of data is just so massive to be able to store the data fast enough even in the biggest databases. Neither streaming in-memory engines (due to their local view) nor off-line databases (due to their non real-time character) are feasible solutions for real-time streaming applications.

A stream processing platform provides the necessary functionality to write continuous and temporal queries able to perform online and real-time data analysis on top of the streamed data.

The enabling component in stream processing is the Complex Event Processor (CEP), which is in charge of actually processing events. In a networked distributed message-based system, CEP technology enables the ability of creating actionable, situation knowledge by correlating system messages with databases and applications, in real-time or near real-time. When properly adopted CEP can provide an organization with the capability to define, manage and predict events, situations, exceptional conditions, opportunities and threats in complex, heterogeneous networks.

Complex Event Processing (CEP) techniques allow us to correlate events produced by different sources. Complex Event Processor queries the bus looking for complex events sequences representative of the ongoing attack. The sequences to look for are obtained by translating abstract queries by applying details inferred from the threat ontology. Abstract queries refer to generic threats. In order to make such queries executable, the Query Translator needs to retrieve information related to what kind of symptoms a threat produces. The ontology allows

decoupling the threat to be monitored (abstract query) from the events giving evidence of such threat in the specific context (threat ontology). When one or more of the registered queries match a particular sequence of events, the Decision Engine raises an alert, possibly providing diagnostics info.

The CEP platform chosen in this architecture is Borealis [15], as it provides interesting characteristic in terms of Scalability, Tolerance, and Performance. Furthermore in Borealis queries are expressed in a relational algebra and this allows simplifies manipulations needed to make abstract queries executable.

3.3 Ontology-driven query generation

The Decision Engine is the component of the proposed Intrusion Detection System able to correlate events provided by the heterogeneous sensors distributed throughout the network in order to improve performance and accuracy of the detection.

The distributed sensors are able to detect the features representative of an ongoing attack and, consequently, feed events reporting different views of it to the Event Bus. Even though the diversity improves the capabilities to detect an attack, the application of several sensors increases the volume of generated events. The Decision Engine is able to appropriately query the Event Bus in order to obtain only the sequences of events it retains relevant to classify an attack, avoiding to be flooded by non-pertinent events. After that, it correlates the resulting events improving the accuracy of the whole detection.

Such a behaviour has been implemented by adopting an ontology based approach, that relies in: i) an ontology specifying the knowledge the Decision Engine uses while performing the correlation activity, ii) a reasoner able to process such an ontology in order to generate the appropriate queries to the bus and correlate the resulting events.

Ontologies provide powerful constructs that include machine interpretable definitions of the concepts and the relations among them. This enables to formalize a domain-specific knowledge so that an area of interest can be systemically and semantically described. In this case, the domain-specific knowledge includes all concepts and relations among them required to perform a correlation-based detection of attacks and intrusions.

The Detection Engine has been equipped with a reasoner for asserting the statements of the Threat Ontology into a knowledge base (KB) and, therefore, for querying the resulting KB. More specifically, such a reasoner, exploiting concepts and relations defined in the ontology, performs queries to the KB in order to identify, for a specific attack, the sensors able to detect it with the best trustworthiness.

As a result, the Detection Engine generates a sequences of queries to be submitted by the CEP to the bus in order to gather only the events produced by the identified sensors.

After receiving the answer from the bus, the Decision Engine, taking into account the trustworthiness of the sensors and the level of confidence of their detections, correlates the events producing a unique and more accurate alert.

4 Related work

A great extent of work as been made with respect to intrusion detection and security events correlation. As for this paper, related work can be mainly categorized in three areas: data acquisition, events processing, and security events management and correlation.

4.1 Data acquisition

In the recent years, much work has been done to address the emerging need for consistent information flow solutions. The main directions explored by the research initiatives are the flexibility with respect to heterogeneous stream formats, the performance constraints, and the reliability of the stream transformation process. One of the major attempts to achieve these results is PADS (Processing Ad hoc Data Sources.) from AT&T [4]. PADS and the underlying processing library developed by AT&T enable the designer to identify, validate and semantically decompose streams of message flows, extract information based upon user-defined contextual or event based rules, etc.

In [7], an hardware-enhanced parsing is augmented to detect and recover from grammatical errors while extracting useful information. The main application field in their work is intrusion detection. Xambala [8] is a commercial product based on hardware-accelerated grammar processing and includes among its application fields content firewalls, EDI routing and monitoring, financial transaction pre-processing, intrusion prevention, etc.

Some previous work related to the specification and handling of data format includes standards for defining a logical representation of data, from which a physical representation is automatically derived [9], and languages and tools for processing of purely binary data [10][11]. In particular, DataScript [10] is a language to describe and manipulate binary data formats as types. It consists of two components, a constraint-based specification language that uses DataScript types to describe the physical layout of data, and a language binding that provides a simple programming interface to script binary data.

4.2 Event Processing

Data streaming have been studied intensively during the last five years. A substantial amount of work is being developed by the database community to build query engines for continuous and temporal queries to be deployed on stream processing systems. Example of these systems are Aurora [12], StreaMon [13], [14], [15]. Born from the fusion of

Aurora and Medusa, developed at Brandeis University, Brown University and MIT, Borealis [16] uses an evolution of the relational algebra to define queries. BOREALIS has been provided of advance characteristics such as distributed elaboration, fault tolerance, graphical query creation.

Building the infrastructure (middleware and supporting platform) for streaming applications is becoming a flourishing market, mainly in the US; an increasing number of spin-offs are being created and existing companies are expanding product and service lines to exploit this market. Among commercial products, Streambase [17] provides a solution based on a Stream Processing Engine powered by StreamSQL, a stream processing query language derived from the standard SQL query model.

Sometimes data stream processing is distinguished by Complex Event Processing (CEP) as the latter operates on multiple incoming data streams potentially by creating derived events. Most of the data stream processing solutions today available also offer CEP facilities. As an example, Coral8 [18] is an industry-leading solution which is characteristic for its easiness of deployment and integration. Query can be expressed in a powerful SQL like language allowing, for example, to describe state machine automata which evolves based on events recognized on the events bus. When the automata reaches a final state it corresponds to the recognition of a complex sequence of events and triggers reaction actions.

4.3 Security events management and correlation

Recently the adoption of CEP solutions has been suggested in the field of security management. A Security Event Manager (SEM) can be defined as “a computerized tool used on enterprise data networks to centralize the storage and interpretation of logs, or events, generated by other software running on the network”. Tim Bass from his blog (The Complex Event Processing Blog) has analyzed what are the key requirements a SEM solution should provide and how existing solutions fail to meet them while CEP has the potentiality to improve SEM solutions.

As for the correlation process and logic, a plenty of work is available. A good review of existing approaches is made in [19] where related literature is reviewed and organized in four main categories: (1) approaches based on similarity between alert attributes; (2) techniques based on predefined attack scenarios; (3) methods based on prerequisites (i.e., pre-conditions) and consequences (i.e., post-conditions) of attacks; and (4) approaches based on multiple information sources.

5 Conclusion and future work

Correlation of attack symptoms from diverse information sources has been proven to be an effective solution to improving Intrusion Detection Systems. In this paper we have discussed issues coming from the adoption of such an

approach and we have illustrated an architecture allowing to fight against such issues. The proposed architecture has been designed as a specialization of the PHDS middleware or a security and safety domain. It is based on the exploitation of correlation of security related information gathered at different points and at different architectural levels of the monitored infrastructure. At the core of the architecture there is a Complex Event Processor which allows real-time processing and correlation of collected data. We have discussed technological implications of the proposed architecture and presented technologies chosen to implement it.

5 Acknowledgments.

This paper describes the results of a research activity conducted within the framework of the research project “A Middleware Infrastructure for Real-Time Processing of Heterogeneous Data Streams (PHDS)”. Authors wish to thank the technical staff of all partners participating in the PHDS project (namely: Synclab, Intesis Spa, HAL Software Srl, CINI, and ICAR): their contribution was key for properly steering the direction of the research activity, as well as for improving the quality of this paper.

References:

- [1] B. Krishnamurthy and J. Wang, On network-aware clustering of web clients, *in ACM Proceedings of SIGCOMM 2000*, 2000.
- [2] C. Cortes, K. Fisher, D. Pregibon, A. Rogers, and F. Smith. Hancock, A language for analyzing transactional data streams, *ACM Trans. Program. Lang. Syst.*, vol. 26, n. 2, pp. 301-338, 2004.
- [3] Paulo Esteves Verissimo, Nuno Ferreira Neves, Miguel Pupo Correia, Intrusion-Tolerant Architectures: Concepts and Design, *in Architecting Dependable Systems*, R. Lemos, C. Gacek, A. Romanovsky (eds.), LNCS 2677, Springer Verlag, 2003.
- [4] Fisher and R. Gruber, PADS: a domain-specific language for processing ad hoc data, *in Proceedings of the ACM SIGPLAN 2005 Conference on Programming Language Design and Implementation*, 2005.
- [5] V. Kodaganallur, Incorporating Language Processing into Java Applications: A JavaCC Tutorial, *IEEE Software*, vol. 21, no 4, pp. 70-77, 2004.
- [6] Campanile, Ferdinando; Cilaro, Alessandro; Coppelino, Luigi; Romano, Luigi, Adaptable Parsing of Real-Time Data Streams, *Parallel, Distributed and Network-Based Processing*, 2007. PDP '07. 15th EUROMICRO International Conference on , vol., no., pp.412-418, 7-9 Feb. 2007
- [7] J. Moscola, Y. H. Cho, J. W. Lockwood, Reconfigurable Context-Free Grammar based Data Processing Hardware with Error Recovery, *in Proceedings of International Parallel & Distributed*

Processing Symposium (IPDPS/RAW), Rhodes Island, Greece, 2006

- [8] Xambala, Using AnyContent™ Semantic Processing to provide a Better Regex Solution, 2005, <http://www.xambala.com/> (last accessed 18/02/2009)
- [9] O. Dubuisson, ASN.1: Communication between heterogeneous systems, *Morgan Kaufmann*, 2001.
- [10] G. Back, “DataScript - A specification and scripting language for binary data”, in *Proceedings of Generative Programming and Component Engineering*, vol. 2487. LNCS, pp. 66-77, 2002.
- [11] P. McCann and S. Chandra, PacketTypes: Abstract specification of network protocol messages, in *ACM Conference of Special Interest Group on Data Communications (SIGCOMM)*, pp. 321-333, 1998.
- [12] D. Carney, U. Çetintemel, A. Rasin, S. B. Zdonik, M. Cherniack, M. Stonebraker: Operator Scheduling in a Data Stream Manager. *VLDB 2003*: 838-849
- [13] S. Babu, J. Widom: StreaMon: An Adaptive Engine for Stream Query Processing. *SIGMOD Conference 2004*: 931-932.
- [14] Utkarsh Srivastava, Kamesh Munagala, Jennifer Widom: Operator placement for in-network stream query processing. *PODS 2005*: 250-258
- [15] Lim, H., Lee, J., Lee, M., Whang, K., and Song, I. 2006. Continuous query processing in data streams using duality of data and queries. In *Proceedings of the 2006 ACM SIGMOD international Conference on Management of Data* (Chicago, IL, USA, June 27 - 29, 2006). SIGMOD '06. ACM, New York, NY, 313-324.
- [16] The Borealis project, <http://www.cs.brown.edu/research/borealis/public/> (last accessed 18/02/2009).
- [17] StreamBase Technical Documentation, available at <http://www.streambase.com> (last accessed 18/02/2009).
- [18] Coral 8, <http://www.coral8.com/> (last accessed 18/02/2009).
- [19] Dingbang Xu and Peng Ning, Correlation Analysis of Intrusion Alerts, *Advances in Information Security, Intrusion Detection Systems, Springer US*, Volume 38, pagg. 65-92 , 2008