Modeling Intrusion Alerts using IDMEF Data Model

Maheyzah Md. Siraj
Faculty of Computer Science and Information System
University Technology of Malaysia
81310 Skudai, Johor, MALAYSIA
Tel : 607 5532245
maheyzah@utm.my

Siti Zaiton Mohd. Hashim
Faculty of Computer Science and Information System
University Technology of Malaysia
81310 Skudai, Johor, MALAYSIA
Tel : 607 5532439
sitizaiton@utm.my

ABSTRACT
In response to proliferated attacks on enterprise systems today, practitioners employ multiple, diverse intrusion detection sensors to improve the detection rate and the coverage within the system for increased information assurance. An important problem in such environment is the management of alerts. One of the essential issues in alerts management is the standardization of the alerts format. For some scholars, such standardization can be referred as alerts normalization. In this paper we address the data model for intrusion detection sensor alerts, called Intrusion Detection Message Exchange Format (IDMEF) and explain the rationale for using this model. An implementation of the data model in the Extensible Markup Language (XML) is presented to represent alerts generated from intrusion detection sensors for better viewing and to ease future alerts analysis for instances aggregation and correlation, regardless of the alerts origin.

Categories and Subject Descriptors
D.4.6 [**Operating Systems**]: Security and Protection–*Invasive software (e.g., viruses, worms, Trojan horses)*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

General Terms
Security

Keywords
Alerts analysis, IDMEF, correlation, intrusion detection, information security

1. INTRODUCTION
Information Assurance and Security (IAS) is viewed as the perception that systems are operating as required with expected protection of the availability, confidentiality and integrity of information within the systems. In order to maintain trust in systems, mechanisms are deployed that monitor any violation of such perception. Intrusion detection sensors[1] have been extensively used by researchers and practitioners to maintain trustworthiness in systems [1,2]. A sensor closely monitors systems and their networks for any sign of probable security violations (e.g., intrusion) and then reports alerts to human analyst to be studied and analyzed [3].

[1] We refer intrusion detection sensor as sensor interchangeably.

The alerts' structure and format may differ depends on the sensors which generated it. Such diversity makes it difficult for human analyst to manage and study the behaviour of attackers. In other words, it is hard to analyze alerts which are not uniformly structured especially in terms of their attributes. Therefore, to be capable of aggregating and correlating alerts for analyzing purposes, it is necessary to adapt or pre-process the alerts log reported by sensors to a common and standard format.

Thus, in this paper we address a data model to represent intrusion alerts in standard and common format called IDMEF. The IDMEF data model is an object-oriented representation of the alert data. The data model is aimed to provide a standard representation of alerts in an unambiguous fashion, and to permit the relationship between simple and complex alerts to be described. Focus of this paper is the standard representation of alerts to facilitate and support the alerts analysis processes at the higher level such as aggregation and correlation.

The rest of this paper is organized as follows. The next section investigates the problems addressed by the IDMEF data model. It is followed by a few related works. Section 4 reviews the IDMEF model and its structure. Section 5 presents concepts and justifications on implementing XML for IDMEF data model. Section 6 describes the experiments to represent alerts in IDMEF as well as its result. Finally, Section 7 concluded this paper and presents some ongoing work.

2. PROBLEMS ADDRESSED BY THE DATA MODEL
The data model addresses several problems associated with representing intrusion alert data [6]. Among them are in the following:

a) *Alert information is inherently heterogeneous.* Some alerts are defined with very little information, such as origin, destination, name, and time of the event. Other alerts provide much more information, such as ports or services, processes, user information, and so on [4,6]. The data model that represents this information must be flexible to accommodate different needs.

b) *Intrusion detection environments are different*. Some sensors detect attacks by analyzing network traffic; others use operating system logs or application audit trail information. Alerts for the same attack, sent by sensors with different information sources, will not contain the same information [5,6]. With the IDMEF data

model defines support classes that accommodate the differences in data sources among sensors. In particular, the notions of source and target for the alert are represented by the combination of Node, Process, Service, and User classes.

c) *Analyzer capabilities are different*. Depending on the environment, one may install a lightweight sensor that provides little information in its alerts, or a more complex sensor that will have a greater impact on the running system but provide more detailed alert information [6]. Therefore, the data model must allow for conversion to formats used by tools other than intrusion detection sensors, for the purpose of further processing the alert information.

d) *Operating environments are different.* Depending on the kind of network or operating system used, attacks will be observed and reported with different characteristics. The data model should accommodate these differences [6].

e) *Commercial vendor objectives are different*. For various reasons, vendors may wish to deliver more or less information about certain types of attacks. The object-oriented approach allows this flexibility while the sub-classing rules preserve the integrity of the model [6].

Most importantly, IDMEF enables interoperability among commercial, open source, and research systems, allowing users to mix-and-match the deployment of these systems according to their strong and weak points to obtain an optimal implementation [6]. As a result, Intrusion Detection and Response Systems (IDRS) research community [5,8-14] has welcomed the use of this format [7].

3. RELATED WORK

Modeling intrusion alerts using IDMEF is an important step to support the fusion of alerts from multiple sensors. Different kind of sensors provided different kind of alerts format especially if they are from different vendors which offered a various types of techniques and system architecture to detect network intrusions.

Debar and Wespi [5] made it clear in their work about correlation that a unified framework for sensor alerts was essential to handle them independently of the source. The IDMEF developed by the Intrusion Detection Working Group (IDWG) was based on this work.

Cuppens and Miege [9] agreed that the format of reported alerts sent by the different sensors should be compliant with the IDMEF format. But in their work, the XML document will be automatically translated into a set of facts and logical predicates. Other related works which implement IDMEF in their initial stages of alerts analysis can also be found in [5,8-13].

4. INTRUSION DETECTION MESSAGE EXCHANGE FORMAT (IDMEF)

The IDMEF is a data model that automated intrusion detection sensors can use to report alerts about events that they believe suspicious. The purpose of the IDMEF is to define data formats and exchange procedures for sharing information of interest to IDRS, and to the management systems that may need to interact with them [6].

The implementation of IDMEF as mentioned in [7] could be useful and beneficial to:

a) a *single Database Management System (DBMS)* that could store the results from a variety of sensors would make it possible for data analysis and reporting activities to be performed on 'the whole picture' instead of just a part of it;

b) an *Alert Correlation System (ACS)* that could accept alerts from a variety of sensors would be capable of performing correlation and calculations;

c) a *Graphical User Interface (GUI)* that could display alerts from a variety of sensors would enable the user to monitor all of the sensors from a single screen, and require the user to learn only one interface, instead of several; and

d) a *Common Data Exchange (CDE)* format would make it easier for different organizations (users, vendors, response teams, law enforcement) to not only exchange data, but also communicate about it.

4.1 IDMEF Data Model

The IDMEF data model is implemented using a Document Type Definition (DTD) to describe XML documents. IDMEF is an object-oriented representation and a Unified Modeling Language (UML) model, which can be summarized as Figure 1 [7] with its class descriptions summarized from [6] in Table 1.

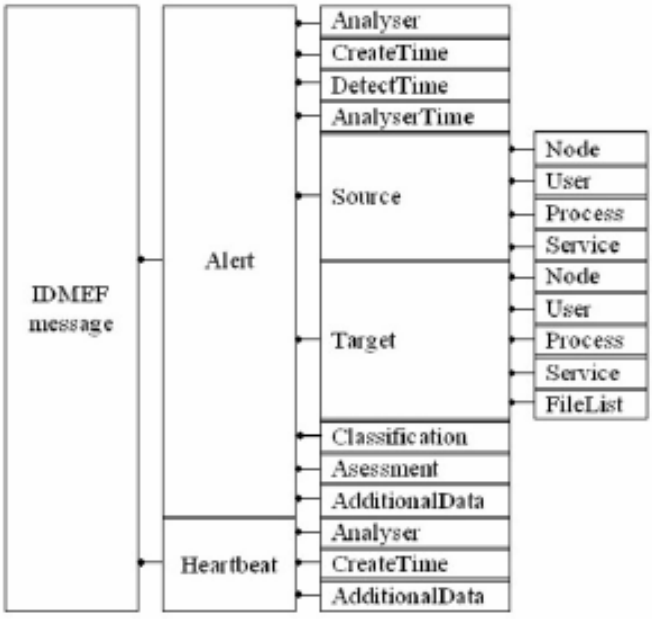

Figure 1. IDMEF data model.

Table 1. The IDMEF-Message Classes.

Class	Description
<Alert>	Depending on the sensor, an *Alert* message may correspond to a single detected event or multiple detected events. Alerts occur asynchronously in response to outside events.
Analyser	Identification information for the analyzer (i.e., sensor) that originated the alert.
CreateTime	The time the alert was created. Of the three times that may be provided with an alert, this is the only one that is required.
DetectTime	The time the event(s) leading up to the alert was detected. In the case of more than one event, the time the first event was detected. In some circumstances, this may not be the same value as *CreateTime*.
AnalyserTime	The current time on the analyzer.
Source	The source(s) of the event(s) leading up to the alert.
Target	The target(s) of the event(s) leading up to the alert.
Classification	The 'name' of the alert.
Assessment	Information about the impact of the event, actions taken by the analyzer in response to it, and the analyzer's confidence in its evaluation.
*AdditionalData	Information included by the analyzer that does not fit into the data model. This may be an atomic piece of data, or a large amount of data provided through an extension to the IDMEF.
<Heartbeat>	Indicate analyzer current status. *Heartbeats* are intended to be sent in a regular period, say, every ten minutes or every hour. The receipt of a *Heartbeat* message from an analyzer indicates that the analyzer is up and running.
Analyser	Identification information for the analyzer that originated the heartbeat.
CreateTime	The time the heartbeat was created.
AdditionalData	Similar description to *.

5. EXTENSIBLE MARKUP LANGUAGE (XML)

The Extensible Markup Language (XML) [15] is a simplified version of the Standard Generalized Markup Language (SGML), syntax for specifying text markup defined by the ISO 8879 standard. XML is gaining widespread attention as a language for representing and exchanging documents and data on the Internet, and as the solution to most of the problems inherent in HyperText Markup Language (HTML). XML was published as recommendation by the World Wide Web Consortium (W3C).

XML is a metalanguage - a language for describing other languages that enables an application to define its own markup. XML allows the definition of customized markup languages for different types of documents and different applications. This differs from HTML, in which there is a fixed set of identifiers with preset meanings that must be 'adapted' for specialized uses. Both XML and HTML use elements (tags) (identifiers delimited by '<' and '>') and attributes (of the form "name='value'")[6].

5.1 Rationale for Implementing IDMEF in XML

XML-based applications are being used or developed for a wide variety of purposes, including electronic data interchange in a variety of fields, financial data interchange, electronic business cards, and many others.

XML's flexibility makes it a good choice for implementing the IDMEF. As discussed in [6], more specific reasons for choosing XML to implement the IDMEF are:

a) *XML allows a custom language to be developed* specifically for the purpose of describing intrusion alerts.

b) *Software tools for processing XML documents are widely available*, in both commercial and open source forms. Numerous tools and APIs for parsing and/or validating XML are available in a variety of languages, including Java, C, C++, Tcl, Perl, Python, and GNU Emacs Lisp. Widespread access to tools will make adoption of the IDMEF by product developers easier, and hopefully, faster.

c) *XML meets IDMEF Requirement 5.1* [16], that message formats support full internationalization and localization. The XML standard requires support for both the UTF-8 and UTF-16 encodings of ISO/IEC 10646 (Universal Multiple-Octet Coded Character Set, UCS) and Unicode, making all XML applications (and therefore all IDMEF-compliant applications) compatible with these common character encodings.

d) *XML meets IDMEF Requirement 5.2* [16], that message formats must support filtering and aggregation. XML's integration with XSL, a style language, allows messages to be combined, discarded, and rearranged. In addition, XML is free, with no license, no license fees, and no royalties.

6. EXPERIMENTS AND DATASET

This paper makes the assumption that all alerts in a particular network session pertain to the same attack. Based on this assumption, the IDMEF representations of all the alerts in each session are put into a single XML document. In effect, this document represents a pattern of alerts that characterize a potential attack.

Performing real attacks in real networks to produce the intrusion alerts as dataset are not realistic [21]. In fact, creating such dataset in the field of analyzing intrusion alerts remains an open research problem [7]. Therefore, we used DARPA 2000 dataset (labeled as LLDOS) which are provided by the Lincoln Lab, Massachusetts Institute of Technology (MIT) [17]. Most of the research community of IDRS evaluated their works with DARPA's dataset. This dataset is, nonetheless, the only publicly available dataset in evaluating IDRSs. In fact there have been a number of publications both in IDRS as well as in the machine learning communities using this dataset for examples in [21-23].

Alerts can be generated using a freely available Network-based Intrusion Detection System software, named *Snort* [18]. A tool

Table 2. An example of an alert in fields.

SensorID	AlertID	SrcIP	DestIP	SrcPort	DestPort	Serv	Time	AlertType
109	289	135.013.216.191	172.016.112.149	22	22	tcp	2007-11-24 17:42:31	STEALTH ACTIVITY

called *tcpreplay* [19] replays the network packets (i.e., *tcpdump* data) before feeding the data through *Snort*. To enable *Snort* read the tcpdump file, *r* option is turned on. Alerts generated were captured and logged in to a file. A *Snort output plug-in* [20] then converted alerts into IDMEF representations (i.e., in XML documents). The hardware specifications used were Pentium 4 processor with 1.50GB RAM.

6.1 Result

For each alert, we only preserve several essential attributes as suggested in [8,9,24] which are: *sensorID, alertID, source IP address, destination IP address, source port, destination port, service name, timestamp and alert type*. To manage all attributes in more manageable way each attributes are stored in table (by fields). An example of an alert attributes in fields is given in Table 2 and its IDMEF representation is showed in Figure 2.

Referring to Table 2 and Figure 2, the alert is uniquely identified by the '*Alert ident*' attribute. The *service* section describes network services on targets. In this case, it contains two attributes, namely *protocol* (tcp) and *port* (22). The target node address is specified by the *target* element and the alert message is given by the *Classification name* attribute. This alert simply reports a *stealth scan* on port 22 from 135.013.216.191 to 172.016.112.149.

The dataset [17] included the network traffic collected from both Demilitarized Zone (DMZ) and the inside part of the evaluation network. The LLDOS scenarios can be divided into five phases as the following:

Phase 1: The attacker scans the network to determine which hosts are alive or 'up'.

Phase 2: The attacker then uses the *ping* option of the *sadmind* exploit program to determine which hosts selected in Phase 1 are running the *sadmind* service.

Phase 3: The attacker attempts the *sadmind Remote-to-Root* exploit several times in order to compromise the vulnerable machine.

Phase 4: The attacker uses *telnet* and *rpc* to install a DDOS program on the compromised machines.

Phase 5: The attacker telnets to the DDOS master machine and launches the *mstream DDOS* against the final victim of the attack.

As illustrated in Table 3, the number of alerts provided from the dataset [17] is enormous. Obviously the least and the most number of alerts are from phase 4 and phase 5 respectively. There is not much intrusion detection in phase 4 because the attacker(s) used a standard protocol (i.e., *telnet*) to establish a connection and run a valid *rpc* to execute the Distributed Denial of Service (DDOS) attack. In contrast, both networks detected a lot of DDOS intrusion from multiple hosts as showed in the last phase.

Figure 2. IDMEF representation of an alert in an XML document.

```
<IDMEF-Message/>
<?xml version="1.0"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFC XXXX IDMEF
v1.0//EN" "/usr/local/etc/idmef-message.dtd">
<IDMEF-Message version="1.0">
   <Alert ident="289">
      <Analyzer analyzerid="109" model="snort" version="2.0.5">
            <Node>
               <name>tcpdump_dmz</name>
            </Node>
      </Analyzer>
      <CreateTime ntpstamp="0xc36cc187.0xd3aa9b49">2007-11-
            24T17:42:31Z</CreateTime>
            <Source>
            <Node>
               <Address category="ipv4-addr">
                  <address>135.013.216.191</address>
               </Address>
            </Node>
                <Service>
                  <port>22</port>
                     <protocol>tcp</protocol>
                </Service>
            </Source>
            <Target>
            <Node>
               <Address category="ipv4-addr">
                  <address>172.016.112.149</address>
               </Address>
            </Node>
                <Service>
                  <port>22</port>
                     <protocol>tcp</protocol>
                </Service>
            </Target>
      <Classification origin="vendor-specific">
            <name>msg=(spp_stream4) STEALTH ACTIVITY (NULL scan)
            detection</name>
               <url>none</url>
      </Classification>
   </Alert>
</IDMEF-Message>
```

Table 3. Alerts frequency for 'DMZ' and 'Inside' network.

Phase	LLDOS1.0_DMZ		LLDOS1.0_Inside	
	# of alerts	Rate (%)	# of alerts	Rate (%)
1	785	2.31	31	0.09
2	24	0.07	32	0.09
3	80	0.24	35	0.10
4	19	0.06	22	0.07
5	33 948	97.32	33 787	99.65
Total	34 851	100	33 907	100

This shows that the total number of alerts overwhelm the human analyst. This situation makes it hard for them to understand, analyze and manage the increasingly overwhelming alerts. Moreover, manually manage them is tedious, time-consuming and error prone [1,4,5,9,22-26]. In practice such case caused by a large number of false positives and redundant alerts [2,27,28]. Therefore, this result encourages us to proceed with the next stages of experiment, which to improve the quality of alerts and to extract the attacker(s) strategies via aggregation and correlation method.

7. CONCLUSION

When analyzing the alerts reported by intrusion detection sensors, the first problem to solve is the diversity of formats used. Therefore, to be capable of aggregate and correlate alerts, it is necessary to adapt or pre-process the messages reported by sensors to a common and standard format, which is known as IDMEF. In this paper, we explore the IDMEF data model to represent intrusion alerts in a common and standard format. This operation is essential in the early stage of correlation processes which aimed to study the behaviour of attacker(s). The dataset generate a huge number of alerts and overwhelm human analyst. As a consequence, human analyst is hardly to understand, analyze and manage the enormous alerts. Therefore, this situation encourages us to continue with our ongoing research which concentrates on alert aggregation and correlation to improve the alerts quality and to automatically extract the attacker(s) strategies.

8. ACKNOWLEDGEMENTS

This paper is based upon study sponsored by Ministry of Higher Education (MOHE), Malaysia.

9. REFERENCES

[1] Kabiri, P. and Ghorbani, A. A. 2005. Research on Intrusion Detection and Response: A Survey. International Journal of Network Security, Vol.1, No.2, 84–102.

[2] Manganaris, S., Christensen, M., Zerkle, D., and Hermiz, K. 2000. A Data Mining Analysis of RTID Alarms. Journal of Computer Networks, 34, 571–577.

[3] Ning, P., Cui, Y., Reeves, D. and Xu, D. 2004. Techniques and Tools for Analyzing Intrusion Alerts. ACM Transactions on Information and System Security (TISSEC), 2, 274-318.

[4] Perdisci, R., Giacinto, G. and Roli, F. 2006. Alarm Clustering for Intrusion Detection Systems in Computer Networks. Engineering Applications of Artificial Intelligence, 19, 429-438.

[5] Debar, H. and Wespi, A. 2001. Aggregration and Correlation of Intrusion-Detection Alerts. In Proceedings of the 4th International Symposium on Recent Advances in Intrusion detection (RAID), Springer Verlang,California, USA, 85-103.

[6] The Intrusion Detection Message Exchange Format (IDMEF), ftp://ftp.rfc-editor.org/in-notes/rfc4765.txt

[7] Zurutuza, U. and Uribeetxeberria, R. 2004. Intrusion Detection Alarm Correlation : A Survey. In Proceedings of the IADAT International Conference on Telecommunications and Computer Networks, December 1-3, 2004.

[8] Cuppens, F. 2001. Managing Alerts in a Multi-intrusion Detection Environment. 17th Annual Computer Security Applications Conference, December 2001. USA : New Orleans. 22–31.

[9] Cuppens, F. and Miege, A. 2002. Alert Correlation in a Cooperative Intrusion Detection Framework. In Proceedings of the IEEE Symposium on Security and Privacy. 202-215.

[10] Carey, N., Clark, A. and Mohay, G. 2002. IDS Interoperability and Correlation Using IDMEF and Commodity Systems. ICICS 2002, LNCS 2513. 252-264.

[11] Ning, P., Cui, Y. and Reeves, D. S. 2002. Analyzing Intensive Intrusion Alerts via Correlation. In Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), Zurich, Switzerland. 74-94.

[12] Valeur, F., Vigna, G. and Kruegel, C. 2004. A Comprehensive Approach to Intrusion Detection Alert Correlation. IEEE Transactions on Dependable and Secure Computing, Vol. 1, No. 3, 146-169.

[13] Long, J., Schwartz, D. and Stoecklin, S. 2006. Distinguishing False from True Alerts in Snort by Data Mining Patterns of Alerts. SPIE Defense and Security Symposium 2006, Orlando (Kissimmee), Florida, April 17-21, 2006.

[14] Stakhanova, N., Basu, S. and Wong, J. 2007. A Taxonomy of Intrusion Response Systems. International Journal Information and Computer Security, Vol. 1, No. 1/2, 169-184.

[15] Sperberg-McQueen, C., Paoli, J., Maler, E., and Bray, T. 2000. Extensible Markup Language (XML) 1.0 (Second Edition. World Wide Web Consortium FirstEdition, http://www.w3.org/TR/2000/REC-xml-20001006

[16] Wood, M. and Erlinger, M. 2007. Intrusion Detection Message Exchange Requirements. RFC 4766, http://www.rfc-archive.org/getrfc.php?rfc=4766

[17] Lincoln Lab, MIT. 2000. DARPA 2000 Intrusion Detection Evaluation Datasets, http://ideval.ll.mit.edu/2000index.html

[18] Snort, 2007. A Lightweight Intrusion Detection for Networks. http://www.snort.org/dl/

[19] Tcpreplay, http://tcpreplay.synfin.net/trac/

[20] Poppi, S. 2004. Snort IDMEF output Plug-In, http://sourceforge.net/projects/snort-idmef/

[21] Pietraszek, T. 2006. Alert Classification to Reduce False Positives in Intrusion Detection. PhD Thesis. Albert-Ludwigs-Universit¨at Freiburg im Breisgau, Germany.

[22] Ning, P., Cui, Y., Reeves, D. and Xu, D. 2004. Techniques and Tools for Analyzing Intrusion Alerts. ACM Transactions on Information and System Security (TISSEC), 2, 274-318.

[23] Maggi, F. and Zanero, S. 2007. On the Use of Different Statistical Tests for Alert Correlation. Short Paper. LNCS 4637. Springer-Verlag Berlin Heidelbeirg. 167-177.

[24] Qin, X. (2005). A Probabilistic-based Framework for INFOSEC Alert Correlation. PhD Thesis. Georgia Institute of Technology, USA.

[25] Zhu, B. and Ghorbani, A. A. 2005. Alert Correlation for Extracting Attack Strategies. International Journal of Network Security, Vol. 3, No. 2, 259-270.

[26] Sheyner, O., Haines, J., Jha, S., Lippmann, R. and Wing, J. M. 2002. Automated Generation and Analysis of Attack Graph. Proceedings of the 2002 IEEE Symposium on Security and Privacy (S&P'02), Oakland, California, May 2002.

[27] Yu, D. and Frincke, D. 2007. Improving the Quality of Alerts and Predicting Intruder's Next Goal with Hidden Colored Petri-Net. Computer Networks 51. 632–654.

[28] Julisch, K. 2003. Using Root Cause Analysis to Handle Intrusion Detection Alarms. PhD Thesis, University of Dortmund, Germany.