Intrusion Detection Interoperability and Standardization

Pravin Kothari
GSEC Assignment Version 1.3
February 19th, 2002

1. Abstract

Intrusion detection is an area of increasing attention and its deployment has accelerated rapidly in enterprises and mission-critical systems over the last few years. Commercial vendors and the open source community have responded with a plethora of intrusion detection products. Now a new issue has surfaced - there is no standard way for these closed and incompatible systems to communicate. Lack of standards hampers research and deployment of intrusion detection technology. First "Common Intrusion Detection Framework" (CIDF) and then simpler to use "Intrusion Detection Message Exchange Format" (IDMEF) have been proposed as the standards to be used by such systems to interoperate and exchange messages. This paper presents the motivation for such standardization efforts and an overview of a potential standard – IDMEF along with its communication protocol IDXP.

2. Introduction to IDS and Its Diversity

IDS stands for Intrusion Detection System, which is available as a software and hardware appliance for detecting inappropriate, incorrect, or anomalous activities including network attacks by hackers over the Internet and within the organization. IDSs that operate on a host, i.e., computers running Windows, Solaris, Linux, etc., to detect malicious activity on that host are called host-based IDSs, and IDSs that operate by sniffing network packets are called network-based IDSs. IDSs are also available for monitoring applications, proxies, etc.

The second split in IDS design is the approach to specify and detect intrusions. The most common approaches to intrusion detection are signature based detection and statistical anomaly detection. Signature matching IDSs are based on recognizing patterns of known exploits. They have low false-alarm rates, but are limited to recognizing configured exploits only, and often report after a system has been compromised. On the other hand, anomaly-detecting IDSs can detect novel exploits and provide an early warning when an attack is underway, both at the cost of high false-alarm rates, often so high that users are overwhelmed by alerts and start disregarding the IDS.

3. Emergence of Meta-IDS & Enterprise Security Managers

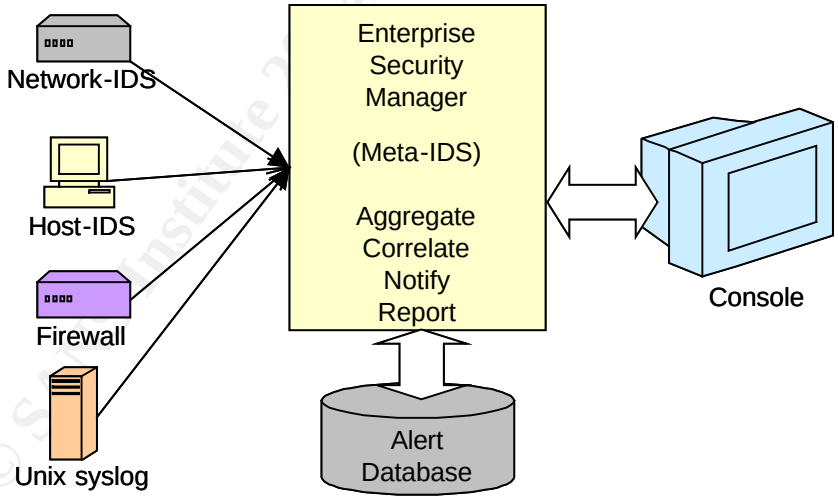
Enterprises typically have at least one IDS on each network and critical host. They must have multiple IDSs at their disposal because the various IDSs all have different strengths and weaknesses based on the characteristics mentioned above, such as, host and network-based. With the current state of the technology and a huge spectrum of potential intrusions, it appears that no single IDS vendor can address the security needs of a typical enterprise.

Due to lack of an industry standard for reporting intrusion incidents, it has become a middleware nightmare to manage the outputs from IDSs of multiple vendors. Monitoring and analyzing alerts from even a handful of IDSs can quickly overwhelm a security staff.

One answer is “Meta-IDS,” [16] or enterprise-level security management console, capable of keeping up with the reams of alert data flowing from security devices including host and network IDSs. The primary role of a Meta-IDS is to ‘aggregate’ - accept security alerts from all deployed IDSs, massage the raw data, extract useful information and present that information in a unified format. It can also ‘cross-correlate’ alerts coming from multiple IDSs, to provide a complete analysis of what attacks are happening across the enterprise network. It’s possible for such a centralized system to eliminate some of the false alarms based on the global knowledge of the enterprise network and hosts. Finally, Meta-IDS can notify a security staff about potential intrusions and provide forensic reporting on the aggregated data.

Adding security devices other than IDSs to the mix, such as, firewall logs, router logs, OS event logs, application security logs, anti-virus logs, etc., and you have just turned a Meta-IDS into a Console for enterprise wide security monitoring. Aggregating all security logs at one centralized location enables enterprises to apply effective correlation techniques in real-time and forensic modes. The more information you can bring to a centralized place, the more effective the correlation becomes in order to detect intrusions that would go undetected otherwise and to reduce false-alarm rates.

IDS vendors (Cisco, ISS, etc.) are making it easier to correlate data from their own sensors, but their products are long way from taking data from firewall, routers, or applications that might have information about the attacks. Enterprises wishing to correlate data cross-device and/or cross-vendor must turn to enterprise security management products such as ArcSight, eSecurity, netForensics and others, or build the solution in-house. For now, these Meta-IDS and enterprise security management products come with their own array of ‘agents,’ each of which interprets alerts coming from a specific security device.



In order to aggregate and correlate, the Meta-IDS or Enterprise Security Manager must be able to receive and interpret alerts from all deployed security devices - even those from different vendors, none of which may use the same format or vocabulary for reporting possible attacks. While no uniform guidelines exist today, there are communities working on standards that will let IDSs speak to each other, as well as to security consoles.

This means Meta-IDS consoles will soon accept input from any IDS or security device that conforms to the standards. In other words, customers will no longer be restricted to correlation

consoles from their IDS vendors or restricted to a set of IDS supported by deployed Meta-IDS. They can acquire best-of-breed IDS for their environment while retaining their preferred Enterprise Security Manager for correlation.

4. Intrusion Detection Standardization Efforts

Among the most extensive of the proposals for interoperability towards a standard intrusion detection protocol are the Common Intrusion Detection Framework (CIDF) and the Intrusion Detection Message Exchange Format (IDMEF).

A. CIDF

CIDF [11] was an attempt by the US government's Defense Advanced Research Projects Agency (DARPA) to develop an IDS interchange language in response to their concerns that no single IDS vendor can address the entire spectrum of attacks. CIDF was a research project and not intended as a standard that would influence the commercial marketplace.

CIDF contains a high-level model consisting of event generators, analyzers, databases, and responders. CIDF specifies a Lisp-like language - Common Intrusion Specification Language (CISL) that is used to communicate between the components. The CISL syntax employs nested S-expressions with a fairly rich vocabulary to form messages describing attacks. The language includes nouns describing the subjects and objects, and verbs, such as “delete” and “open session”. While quite powerful, some IDS authors have found CISL to be cumbersome, and to date its practical applications have been limited. It has, however, been influential in shaping other efforts in research. CIDF development took place during 1997-99 and some of the ideas that came out of it have been rolled into IDMEF, which is a separate work.

B. IDMEF

The Internet Engineering Task Force (IETF) is the body, which develops new Internet standards. They have a working group, the Intrusion Detection Exchange Format Working Group (IDWG)[1], to develop a common format for IDS alerts. Its charter is “to define data formats and exchange procedures for sharing information of interest to intrusion detection and response systems, and to management systems which may need to interact with them.”

The working group has proposed IDMEF - an XML based specification for intrusion alert format for these systems to communicate. A lot of attention has been paid to the needs of IDS analysis, and to making the protocol work through firewalls in a straightforward way. IDMEF message format is independent of the communication protocol. IDWG has also proposed IDXP as the communication protocol for IDMEF, but it’s not a necessary requirement for IDMEF applications.

IDWG, IDMEF, IDP, IAP, and IDXP, all related and slightly confusing terms, ironically came from the same standardization effort. Here is a quick review to make the rest of the paper clearer. IDWG is the name of the working group, and the more popular term IDMEF is the message format, which is independent of the communication protocol. IDP is the specification (not implementation) for the IDMEF communication protocol described in the same IDMEF requirement specification [3]. There are two implementations of IDP – the early one is called IAP (Intrusion Alert Protocol[6]), which did not make it to the RFC. The newer

and recommended one is IDXP – Intrusion Detection eXchange Protocol[5] – described below in section 9.

5. IDMEF Motivation

Besides the Meta-IDS need described above, the IDMEF requirement specification [3] describes the following additional reasons for such a standard:

Intrusions frequently involve multiple organizations as victims, or multiple sites within the same organization. Typically, those sites will use different IDSs. It would be very helpful to correlate such distributed intrusions across multiple sites and administrative domains. Having reports from all sites in a common format would facilitate this task.

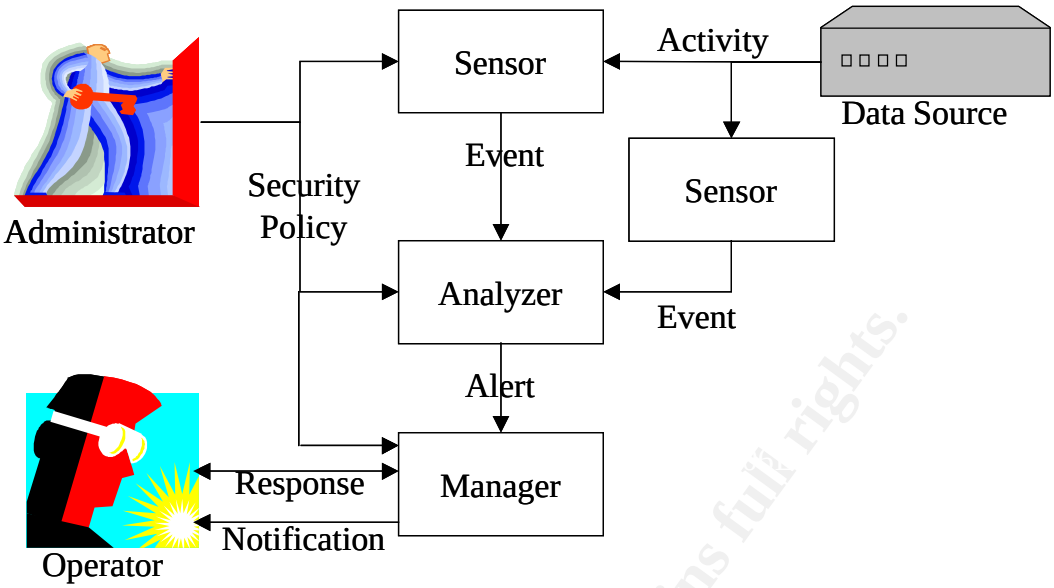
The existence of a common format should allow components from different IDSs to be integrated more readily. Thus, intrusion detection research should migrate into commercial products more easily.

Let's take a scenario of DShield.org [20], which is an attempt to collect data about suspicious activity from all over the Internet. This data is cataloged and summarized to discover simple trends in activities. They require members to submit logs in all sorts of formats by installing specialized 'clients' from a huge array of supported devices. Then they need to re-map different log formats to their central database by losing some of the information. If there were standards for security logs, such as IDMEF, it would have been much easier for members to send logs and also, DShield.org could have efficiently inserted the log into their database and retarget their efforts from writing clients to working on finding complex patterns utilizing complete information.

All of these reasons suggest that a common format for reporting alerts should help the IDS market to grow and innovate more effectively, and should also result in users obtaining better results from deployment of intrusion detection systems.

6. IDMEF Terminology

The activities of the IDWG are integrated closely with Intrusion Detection Systems, so descriptions of the IDMEF use typical IDS terminology. Some of the terms used in IDMEF are described below:



The diagram above illustrates the terms described below and their relationships. Not every IDS will have all of these separate components exactly as shown. Some IDSs will combine these components into a single module; some will have multiple instances of these modules.

Activities: Activities are elements of the data source that are identified by the sensor or analyzer as being of interest to the operator. Examples of this include network session showing unexpected telnet activity, operating system log file entries showing a user attempting to access files to which he or she is not authorized to have access, etc.

Event: Activity that is detected by the sensor and which may result in an IDMEF alert being transmitted. For example, 'N' failed logins in 'T' seconds might indicate a brute-force login attack.

Alert: A message from an analyzer to a manager that an event of interest has been detected. An alert typically contains information about the unusual activity that was detected, as well as the specifics of the occurrence.

Administrator: The human with overall responsibility for setting the security policy of the organization including decisions about deploying and configuring the IDSs.

Operator: The human that is primary user of the IDS manager for initiating responses to alerts and notifications.

Sensor: The ID component that collects data about activity from data sources, detects events, and forwards them to the analyzer.

Analyzer: The ID component that analyses the events and according to the security policy possibly generates alerts based on these events. Alerts are formatted and transferred to managers using the IDMEF format over IDXP (optional) transfer protocol. In many existing IDSs, the sensor and the analyzer are part of the same component.

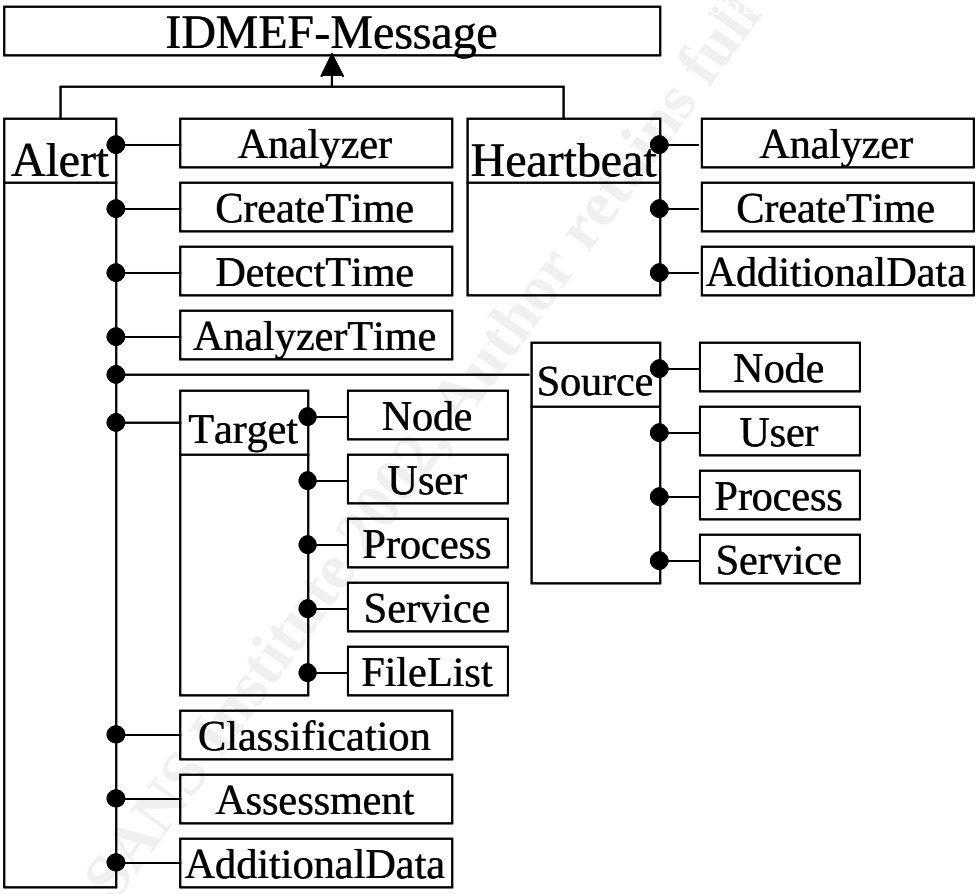
Manager: The ID component or process from which the operator manages the various components of the ID system. Management functions typically include (but are not limited to)

sensor configuration, analyzer configuration, event notification management, data consolidation, and reporting. Managers inform the operator through different types of notification that alerts have occurred, as per the security policy.

7. IDMEF Data Model

The top-level class for all IDMEF messages is IDMEF-Message; each type of message is a subclass of this top-level class. There are presently two types of messages defined: Alerts and Heartbeats. Within each message, subclasses of the message class are used to provide the detailed information carried in the message.

The relationship between the principal components of the data model is shown in the following figure (occurrence indicators and attributes are omitted).



It is important to note that the data model does not specify how an alert should be classified or identified. For example, a port-scan might be identified by one analyzer as a single attack against multiple targets while another analyzer might identify it as multiple attacks from a single source. However, once an analyzer has determined the type of alert it plans to send, the data model dictates how that alert should be formatted.

8. IDMEF Message Content

There are many different types of IDSs, such as those based on: signatures, anomalies, correlation, network monitoring, host monitoring, or application monitoring. IDMEF strives to

accommodate these diverse approaches by concentrating on conveying '**what**' an IDS has detected, rather than '**how**' it detected it.

The content of IDMEF messages contains the identified name of the event (Classification) if it is known. This name is drawn from a standardized list (bugtraq, cve, etc.) of events (if available) or will be an implementation-specific name if the event identity has not yet been standardized. Consider a scenario where an attack is detected by two different analyzers from two distinct implementations. Both report the same event identity to the manager, even though the algorithms used to detect the attack by each analyzer might have been different.

The IDMEF message contains an indication of the possible impact of this event on the target. It also allows references to additional detailed data related to this specific underlying event.

The IDMEF message contains the identity of the source of the event and target component identifier if it is known. In the case of a network-based event, this will be the source and destination IP address of the session used to launch the event. It supports the representation of different types of device addresses, such as, IP address, MAC address, addresses used in ATM switch fabric, etc.

The following classes are some of the core classes in the IDMEF message as described in IDMEF Data Model [4]:

The IDMEF-Message Class: All IDMEF messages are instances of the IDMEF-Message class; it is the top-level class of the IDMEF data model, as well as the IDMEF DTD. There are currently two types of IDMEF-Message: Alert and Heartbeat.

The Alert Class: Generally, every time an analyzer detects an event that it has been configured to look for, it sends an Alert message to its manager(s). Depending on the analyzer, an Alert message may correspond to a single detected event, or multiple detected events. Alerts occur asynchronously in response to outside events. The Alert class has one optional attribute *ident* - a unique identifier for the alert. Alert is represented in the XML DTD as follows:

```
<!ELEMENT Alert                         (
    Analyzer, CreateTime, DetectTime?, AnalyzerTime?, Source*,
    Target*, Classification+, Assessment?, (ToolAlert |
    OverflowAlert | CorrelationAlert)?, AdditionalData*
  )>
<!ATTLIST Alert
    ident               CDATA                   '0'
  >
```

Classification: One or more. The "name" of the alert, or other information allowing the manager to determine what it is.

The Analyzer Class: The Analyzer class identifies the analyzer from which the alert or heartbeat message originates. Only one analyzer may be encoded for each alert or heartbeat, and that MUST be the analyzer at which the alert or heartbeat originated.

The AdditionalData Class: Zero or more. 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.

The Heartbeat Class: Analyzers use Heartbeat messages to indicate their current status to managers. 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 to the manager that the analyzer is up and running; lack of a Heartbeat message (or more likely, lack of some number of consecutive Heartbeat messages) indicates that the analyzer or its network connection has failed.

The CreateTime Class: The CreateTime class is used to indicate the date and time the alert or heartbeat was created by the analyzer.

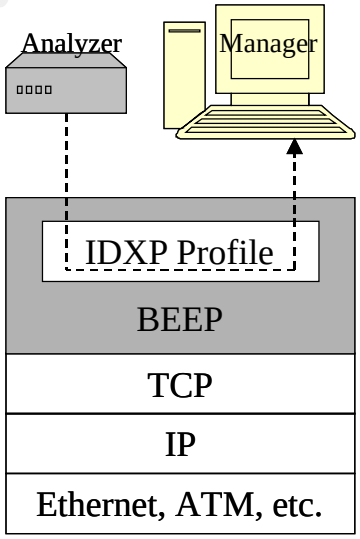
9. IDMEF Communication Protocol - IDXP

Analyzers create and communicate alerts to managers across a TCP/IP network. This communication must be done in a standard and secure fashion and is referred as the IDMEF Communication Protocol (IDP) in the requirement specifications [3]. Note that IDP is a specification and not an implementation.

The IDP specification requires reliable transmission of messages between ID components across firewall boundaries without compromising security. The implementation must also support mutual authentication of the analyzer and the manager to each other. The support is also required for confidentiality of the message content with a variety of encryption algorithms as well as integrity and resistance for DoS attacks.

The working group's first attempt to meet IDP requirements was the development of the Intrusion Alert Protocol (IAP)[6]. The design of IAP was based on HTTP, which turned out to be unsuitable for several reasons.

Then the working group decided to use the Blocks Extensible Exchange Protocol (BEEP)[9], which itself is a new IETF general framework for application protocols. BEEP does not rely on any particular transport protocol. It specifies requirements on how an underlying transport protocol must support a BEEP session. It maps easily to any connection oriented transport protocol offering ordered and reliable delivery including TCP/IP.



BEEP framework is used as the basis to derive an Intrusion Detection Exchange Protocol (IDXP). It allows authentication and confidentiality through the use of 'profiles.' So a profile is

created to specify IDXP requirements in the BEEP framework. A 'Tunnel' profile is also created for exchanging messages through firewalls.

10. IDMEF Examples

The following examples demonstrate how the IDMEF is used to encode alert data.

a. The "teardrop" Attack

Network-based detection of the "teardrop" attack

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFC XXXX IDMEF v1.0//EN" "idmef-message.dtd">
<IDMEF-Message version="1.0">
  <Alert ident="abc123456789">
    <Analyzer analyzerid="hq-dmz-analyzer01">
      <Node category="dns">
        <location>Headquarters DMZ Network</location>
        <name>analyzer01.bigcompany.com</name>
      </Node>
    </Analyzer>
    <CreateTime ntpstamp="0xbc723b45.0xef449129">
      2000-03-09T10:01:25.93464-05:00
    </CreateTime>
    <Source ident="a1b2c3d4">
      <Node ident="a1b2c3d4-001" category="dns">
        <name>badguy.hacker.net</name>
        <Address ident="a1b2c3d4-002" category="ipv4-net-mask">
          <address>123.234.231.121</address>
          <netmask>255.255.255.255</netmask>
        </Address>
      </Node>
    </Source>
    <Target ident="d1c2b3a4">
      <Node ident="d1c2b3a4-001" category="dns">
        <Address category="ipv4-addr-hex">
          <address>0xde796f70</address>
        </Address>
      </Node>
    </Target>
    <Classification origin="bugtraqid">
      <name>124</name>
      <url>http://www.securityfocus.com</url>
    </Classification>
  </Alert>
</IDMEF-Message>
```

b. Simple Port Scanning

Network-based detection of a port-scan. This shows detection by a single analyzer; see the next example for the same attack as detected by a correlation engine. Note the use of <portlist> to show the ports that were scanned.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFC XXXX IDMEF v1.0//EN"
  "idmef-message.dtd">
<IDMEF-Message version="1.0">
  <Alert ident="abc123456789">
    <Analyzer analyzerid="hq-dmz-analyzer62">
      <Node category="dns">
        <location>Headquarters Web Server</location>
        <name>analyzer62.bigcompany.com</name>
      </Node>
    </Analyzer>
    <CreateTime ntpstamp="0xbc72b2b4.0x00000000">
      2000-03-09T15:31:00-08:00
```

```
    </CreateTime>
    <Source ident="abc01">
      <Node ident="abc01-01">
        <Address ident="abc01-02" category="ipv4-addr">
          <address>222.121.111.112</address>
        </Address>
      </Node>
    </Source>
    <Target ident="def01">
      <Node ident="def01-01" category="dns">
        <name>www.bigcompany.com</name>
        <Address ident="def01-02" category="ipv4-addr">
          <address>123.234.231.121</address>
        </Address>
      </Node>
      <Service ident="def01-03">
        <portlist>5-25,37,42,43,53,69-119,123-514</portlist>
      </Service>
    </Target>
    <Classification origin="vendor-specific">
      <name>portscan</name>
      <url>http://www.vendor.com/portscan</url>
    </Classification>
  </Alert>
</IDMEF-Message>
```

c. Correlated Alerts

The following example shows how the port scan alert from previous example could be represented if it had been detected and sent from a correlation engine, instead of a single analyzer.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFC XXXX IDMEF v1.0//EN"
  "idmef-message.dtd">
<IDMEF-Message version="1.0">
  <Alert ident="abc123456789">
    <Analyzer analyzerid="bc-corr-01">
      <Node category="dns">
        <name>correlator01.bigcompany.com</name>
      </Node>
    </Analyzer>
    <CreateTime ntpstamp="0xbc72423b.0x00000000">
      2000-03-09T15:31:07Z
    </CreateTime>
    <Source ident="a1">
      <Node ident="a1-1">
        <Address ident="a1-2" category="ipv4-addr">
          <address>222.121.111.112</address>
        </Address>
      </Node>
    </Source>
    <Target ident="a2">
      <Node ident="a2-1" category="dns">
        <name>www.bigcompany.com</name>
        <Address ident="a2-2" category="ipv4-addr">
          <address>123.234.231.121</address>
        </Address>
      </Node>
      <Service ident="a2-3">
        <portlist>5-25,37,42,43,53,69-119,123-514</portlist>
      </Service>
    </Target>
    <Classification origin="vendor-specific">
      <name>portscan</name>
      <url>http://www.vendor.com/portscan</url>
    </Classification>
    <CorrelationAlert>
      <name>multiple ports in short time</name>
```

```
      <alertident>123456781</alertident>
      <alertident>123456782</alertident>
      <alertident>123456783</alertident>
      <alertident>123456784</alertident>
      <alertident>123456785</alertident>
      <alertident>123456786</alertident>
      <alertident analyzerid="a1b2c3d4">987654321</alertident>
      <alertident analyzerid="a1b2c3d4">987654322</alertident>
    </CorrelationAlert>
  </Alert>
</IDMEF-Message>
```

d. Heartbeat

This example shows a heartbeat message that provides "I'm alive and working" information to the manager. Note the use of <AdditionalData> elements, with "meaning" attributes, to provide some additional information.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFC XXXX IDMEF v1.0//EN" "idmef-message.dtd">
<IDMEF-Message version="1.0">
  <Heartbeat ident="abc123456789">
    <Analyzer analyzerid="hq-dmz-analyzer01">
      <Node category="dns">
        <location>Headquarters DMZ Network</location>
        <name>analyzer01.bigcompany.com</name>
      </Node>
    </Analyzer>
    <CreateTime ntpstamp="0xbc722ebe.0x00000000">
      2000-03-09T14:07:58Z
    </CreateTime>
    <AdditionalData type="real" meaning="%memused">
      62.5
    </AdditionalData>
    <AdditionalData type="real" meaning="%diskused">
      87.1
    </AdditionalData>
  </Heartbeat>
</IDMEF-Message>
```

11. Related Intrusion Detection Initiatives

The Intrusion Detection area has been blessed with several standardization efforts with slow but steady progress. CVE and BugTraq are examples of such efforts. Some of these efforts, especially the IDMEF derived IODEF, are summarized below.

A. IODEF - Incident Object Description and Exchange Format

The purpose of the Incident Object Description and Exchange Format [10] defines a common data format and common exchange procedures for sharing information needed to handle an incident between different CSIRTs (Computer Security Incident Response Teams) and to exchange incident related data that allows both known and new types of incidents to be formatted and exchanged. The IODEF is based on the work done by IDWG and the leverage is helping their efforts to move fast.

The IODEF's message is called IncidentAlert, which embeds the IDMEF message to cover most of the details. IODEF's terminology is a superset of IDMEF's terminology as it includes new terms, such as, Attack, Attacker, Damage, Event, Evidence, Incident, Impact, Victim, Vulnerability, etc. The working group has prototype work done to integrate IODEF with Remedy API.

B. IDIP: Intrusion Detection and Isolation Protocol

Funded by DARPA, the Intruder Detection and Isolation Protocol (IDIP) [21] is an infrastructure for integrating IDSs and automated response components being developed at UC Davis. IDIP provides cooperation among intrusion detection systems, firewalls, routers, network management components, and hosts so that intrusions that cross multiple network boundaries can be automatically traced and blocked as close to the source as possible. IDIP has been tested with a variety of IDSs, firewalls, and host-based responders. It provides a discovery coordinator API to allow components access to services including data management, situation display, access to network management and response policy management. IDIP uses CISL as the attack description language. The emphasis in IDIP is on data management and secure communications between diverse components.

12. IDMEF Weaknesses

XML makes it easier to develop and deploy, but it comes with a performance cost. Parsing XML messages is still a relatively slow task today, which is acceptable enough for order-processing and inter-business applications. Since alerts received from IDSs are usually low, XML should work fine for the purpose. However, firewalls, routers, and such network devices, producing logs hundreds and thousands time higher than IDS alerts, may find IDMEF slow to work with. XML dependency of IDMEF may become a stumbling block for extending IDMEF coverage to such network devices.

Current classification attribute "origin" has limited choices: "unknown", "bugtraqid", "cve" and "vendor-specific". Getting into classifications and content semantics are considered beyond the scope of IDWG, which is a good as well as bad decision. It's good because it provides the needed focus for IDWG efforts, keeps it simple, and also it does not confront them against IDS vendors with conflicting and overlapping rules. However, it's bad because it would cause a glut of classification schemes – every IDS vendor rolling their own "vendor-specific" list since their alerts are not always vulnerabilities found on CVE list.

It appears that IDMEF Data model fits very well with Network IDSs, but it does not map as well with non-IDS devices, such as, firewalls, NT Event Log, syslog, etc. For that, IDWG has created a 'catch-all' – AdditionalData class, to hold analyzer supplied information that does not fit into the data model. For instance, Checkpoint OPSEC interface produces over a dozen fields that are missing in the data model that will end up in AdditionalData. Since AdditionalData information goes as hidden details rather than top-level objects, most applications may not be able to manage and operate on it properly. Correlation will become less effective when interesting pieces of information are hidden in AdditionalData with different tag-names from different devices.

Lastly, IDXP is based on BEEP, which itself is a new standard, so IDXP implementation may take long to become feasible for commercial deployment.

13. IDWG Status and Extensions

IDWG has delivered the following 4 documents that existed as Internet Drafts for a while. They are finishing the "final-call" period in mid-February 2002, and then these documents will be forwarded to the Internet Engineering Steering Group (IESG) for publication as RFCs.

Intrusion Detection Message Exchange Requirements [3]
IDMEF Data Model and XML DTD [4]
The Intrusion Detection Exchange Protocol (IDXP) [5]
The TUNNEL Profile[7]

As intrusion detection systems evolve, the IDMEF data model and DTD will have to evolve along with them. To allow new features to be added as they are developed, both the data model and the DTD can be extended. There are two mechanisms for extending the IDMEF data model, inheritance and aggregation. And for extending the XML DTD, the AdditionalData class allows implementers a 'catch-all' to include arbitrary data items in an Alert or Heartbeat message.

IDWG's work will be RFC soon, so IDWG will need to decide what to do next. It could be as simple as working on the next version of document set or join the IODEF efforts, standardize vulnerability alerts, etc. The job is far from over though, as the real challenge is to get the industry acceptance and that may involve taking the work to the next level and showing a few successful deployments.

14. Industry Acceptance

IDWG has successfully attracted participation of several industry leaders, such as, Cisco, NAI, HP, Boeing, IBM, ISS, MITRE, MSFT, Nokia, etc.

It still remains to be seen whether IDMEF and IDXP will receive acceptance from commercial IDS vendors. Standards always take longer than expected for wide acceptance and IDWG is no exception. Once IDWG drafts become official RFC documents, we expect to see implementation from some of the IDS vendors, especially the ones who were involved with IDWG.

The evolving correlation technologies will rely on such standards. Meta-IDS and enterprise security management vendors should adopt IDMEF early. IBM's Tivoli Risk Manager and ArcSight's Enterprise Security Manager are early adopters of IDMEF with demonstrated implementations. Another enterprise console vendor to announce support for IDMEF is eSecurity. Growth in deployment of Enterprise Security solutions in enterprises should also drive up the demand for such standards and interoperability.

There are several tools and libraries available to help build IDMEF and IDXP applications. Most of them are accessible from Silicon Defense's IDWG Web page [2]. Silicon Defense, a security research and consulting company, is actively involved with IDWG. It has also delivered a free open-source library and a plug-in to enable SNORT to output IDMEF XML alerts, which seems to be the most popular implementation available for IDMEF today.

IDMEF is also being used in several research projects. See references [12], [13] and [14] for links to some of them.

15. Conclusions

Lack of open standards for exchanging intrusion alerts and lack of interoperability could potentially hamper the growth of IDS deployment and research. Although IDMEF and IDXP have not yet been blessed as standard, they are very close to becoming one and have already started to get traction with research community, open source, and derivative standardization

efforts. However, further work and success stories may be required to convince IDS vendors and wider intrusion detection community of its usefulness.

What HTML and HTTP did for Internet growth, IDMEF and IDXP can make similar impact on research and deployment of intrusion detection technology. These are the steps in the right direction and we must collaborate to take it further and to keep ahead of the hacker community.

16. References

[1] "Intrusion Detection Exchange Format (IDWG)." Official Working Group Web page. URL: http://www.ietf.org/html.charters/idwg-charter.html (18 Feb. 2002).

[2] "IDWG." at Silicon Defense. URL: http://www.silicondefense.com/idwg/ (18 Feb. 2002).

[3] Wood, M. and Erlinger, M. "Intrusion Detection Message Exchange Requirements." Internet-Draft: work in progress. Feb. 2002. URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-requirements-06.txt (18 Feb. 02).

[4] Curry, D., and Debar, H. "Intrusion Detection Message Exchange Format Data Model and DTD." Internet-Draft: work in progress. Dec. 2001. URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-06.txt (18 Feb. 2002).

[5] Feinstein, B., Matthews, G., and White, J. "The Intrusion Detection Exchange Protocol (IDXP)." Internet-Draft: work in progress. Jan. 2002. URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-beep-idxp-04.txt (18 Feb. 2002).

[6] Buchheim, T., Feinstein, B., Gupta, D., Matthews, G., and Pollock, R. "IAP: Intrusion Alert Protocol." Internet-Draft: work in progress. Mar. 2001. URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-iap-05.txt (18 Feb. 2002).

[7] New, D. "The TUNNEL profile." Internet-Draft: work in progress. Aug. 2001. URL: http://www.ietf.org/internet-drafts/draft-ietf-idwg-beep-tunnel-02.txt (18 Feb. 2002).

[8] Buchheim T., Erlinger M., Feinstein B., Pollock R., Matthews G., Betser J., and Walther A. "Implementing the Intrusion Detection Exchange Protocol." A paper from 17th Annual Computer Security Applications Conference. Dec. 2001. URL: http://www.acsac.org/2001/papers/67.pdf (18 Feb. 2002)

[9] Rose, M. "The Blocks Extensible Exchange Protocol core." RFC 3080. Mar. 2001. URL: http://www.ietf.org/rfc/rfc3080.txt (18 Feb. 2002).

[10] "IODEF - Incident Object Description and Exchange Format Working Group" URL: http://www.terena.nl/task-forces/tf-csirt/iodef/ (18 Feb. 2002).

[11] "Common Intrusion Detection Framework (CIDF)." URL: http://www.isi.edu/~brian/cidf/ (18 Feb. 2002).

[12] Reilly, M., and Stillman, M. "Open Infrastructure for Scalable Intrusion Detection." A paper from 1998 IEEE Information Technology Conference. URL: http://www.oracorp.com/projects/publications/openInfrastructures.doc (18 Feb. 02).

[13] Goldman R., Heimerdinger W., Harp S, Geib C., Thomas V., and Carter R. "Information Modeling for Intrusion Report Aggregation." URL: http://www.geocities.com/rpgoldman/papers/discex01irm.pdf (18 Feb. 2002).

[14] Vigna G., Kemmerer, R. and Blix, P. "Designing a Web of Highly-Configurable Intrusion Detection Sensors." UCSB. URL: http://www.cs.ucsb.edu/~rsg/pub/2001_vigna_kemmerer_blix_raid01.ps.gz (1 Feb. 2002).

[15] "What open standards exist for Intrusion Detection?" URL: http://www.sans.org/newlook/resources/IDFAQ/ID_standards.htm (18 Feb. 2002).

[16] Loshin, P. "New Directions in Intrusion Detection - 'Meta' Detection." Information Security August 2001 (2001). URL: http://www.infosecuritymag.com/articles/august01/cover.shtml (18 Feb. 2002).

[17] George, H. Intrusion Detection - Systems for Today and Tomorrow, SANS Paper. URL: http://rr.sans.org/intrusion/tomorrow.php (18 Feb. 2002).

[18] Northcutt, S., McLachlan, D., and Novak, J. Network Intrusion Detection: An Analyst's Handbook . (Ch 10: Interoperability and Correlation)

[19] Lee, W. Recent Advances in Intrusion Detection : 4th International Symposium, Raid 2001, Davis, CA, USA, October 2001 Proceedings (Lecture Notes by Wenke Lee (Editor), Ludovic Me (Editor), Andreas Wespi (Editor)

[20] "DShield – Distributed Intrusion Detection System." URL: http://www.dshield.org/ (18 Feb. 2002).

[21] "Summary of the Intruder Detection and Isolation Protocol (IDIP) Project." URL: http://seclab.cs.ucdavis.edu/projects/idip.html (18 Feb. 2002).