Incident Response Database

by Erika Shehan, Patrick Fitzgerald, Brian Poole,
Matthew Wirges and Pascal Meunier

Goals:

- Gather statistics
  - Costs by type of incident
    - billing
    - security survey
    - insurance
  - Number of incidents
    - by type
    - by operating system
    - by application
    - by vulnerability exploited

- Support incident Response
  - File upload
  - Email archive
  - Change log
  - Classification
  - Assignment (ownership)
  - Compartmentalization by domains
  - Escalation
  - Information sharing and cooperation
  - Opening and closing incidents

How to classify incidents?

- Need to classify incidents as one category or type
- Different from classifying vulnerabilities
- Different from describing incidents
  - Not a language as in (Howard and Longstaff, Sandia Labs report)
- But, multifaceted nature
  - Recon
  - Many attacks
    - many events per attack
  - Unpredictable duration
  - Obfuscation
- Graceful convergence desirable
  - Dynamic classification
  - Incomplete data
  - New data uncovered in process
  - Incident may be in progress
- Disregard the identity of attacker because it is too difficult to observe

Worst Threat (Ultimate Impact)

What is the worst thing that happened?

- For Who?
  - Administrator
  - User
  - Client/relation of user
- Using which criteria?
  - Monetary value
  - Absolute scales
- Function of the target
- Compare only things in similar categories
  - Integrity
  - Confidentiality
  - Physical threats
  - Accessibility
- Define a “Max” function for each category
- New data on incident:
  - Max(Old, New) for each category
  - Graceful convergence
- Classification of merged incidents/new data
  - Max(Inc1, Inc2)

Dominating Threat Taxonomy

- What threat makes all others below in a hierarchy possible or irrelevant?
  - Theft of computer/storage dominates a denial-of-service attack
  - Denial-of-service attack dominates a recon (scan)
  - Root access dominates normal user access, or read access through a web server vulnerability

- Point of view doesn't matter for single target

- For a single target,
  - Objectivity
  - Determinism
  - Repeatability
  - Specificity
  - Graceful convergence

- Multiple targets
  - Assume equal importance of targets
  - Or most important target must be identified in the description of the incident -- may not be objective
  - Or use one incident/target

CIRDB Hierarchical Domain System

Flat Domain Space:

- Fast searches
- Simple permissions
  - Provides Confidentiality

Hierarchical Domain Space:

- Suitable for incident escalation
- Better report generation
- Allows you to model your IRDB domains around your organizational structure.
- Allows you to tailor your privileges for your organizational structure
- Incident Merging

Implementation of Hierarchical Domain Space

- One central root domain
- Domain space takes on a ‘tree’ structure
- Each domain has ‘parent’ pointer for its location in the tree
- Permissions can be set for individual domains
- Permissions are available to user from any domain below and including the permission origin domain
- Each domain has entries corresponding to each domain it belongs to for easy and quick searches/lookups
    - Database access is faster than an iterative search in large domain situations

User Interface with Hierarchical Domain Space

- Horizontal Navigation – Access domains with a common parent.
- Vertical Navigation – Traverse the domain ‘tree’ within the user’s permission bounds
- Jump Points – Quick access to all domains where permissions originate
- Multi-Incident Views
  - View Incidents for current domain
  - View ‘Dominated Incidents’– All domains below and including the current domain
- Auto updated permissions

Incident Escalation

Sometimes a user may need to pass along an incident to someone with higher privilege. With incident escalation the necessity for a hierarchical domain space becomes more evident. Instead of incidents remaining in their own secluded domain, incidents can now be pushed or escalated to their parent domain for a more privileged user to work with.

Privilege Inheritance

Users may inherit their privileges from parent domains when available. For example if we have a domain "A" and domain "B" (where "A" is the parent domain of "B") and user *x* has Administrator privileges in "A", then user *x* may be able to perform Administrator operations on any incident in domain "B" regardless of who has created it.

Model Organization Structure

Owners may easily model their IRDB system around their entities' structure. Each department could have a domain under the entities root domain and each department can setup up their child domains to fit their individual structure as well.

Incident Merging

An administrator may wish to merge one or more related incidents under his/her control. The hierarchical domain space allows the Administrator[1] to take any incidents in any child domains and merge them into a single incident in their highest domain.

[1] Only an Administrator can perform this function.

Intrusion Detection Message Exchange Format

The IRDB automatically accepts data from intrusion detection systems adhering to the new Internet Engineering Task Force standard proposed by the Intrusion Detection Exchange Format Working Group (IDWG). IDMEF is a standard format automated intrusion detection systems can use for reporting what they have deemed to be suspicious or of interest.

Why have a standard format?

- A standard format enables interoperability among commercial, open source, and research systems. Users can deploy multiple systems according to their strong and weak points to obtain an optimal implementation.

- A standard format for reporting suspicious activity should help the intrusion detection systems market to grow and innovate more successfully, which will result in users obtaining better results from deployment of intrusion detection systems.

- A standard format makes it easier for different organizations, such as users, vendors, response teams, and law enforcement to not only exchange data, but also communicate about it.

IDMEF and XML

- IDMEF calls for using the Extensible Markup Language (XML), a language for describing other languages.

- XML is ideal for the IDMEF, as it allows for the creation of a custom language for describing alerts.

- Meets IDMEF requirements 5.1 and 5.2. Using XML, message formats support full internationalization and localization, as well as filtering and aggregation.

- Tools for processing XML are widely available

- XML is free--no licenses, fees, or royalties

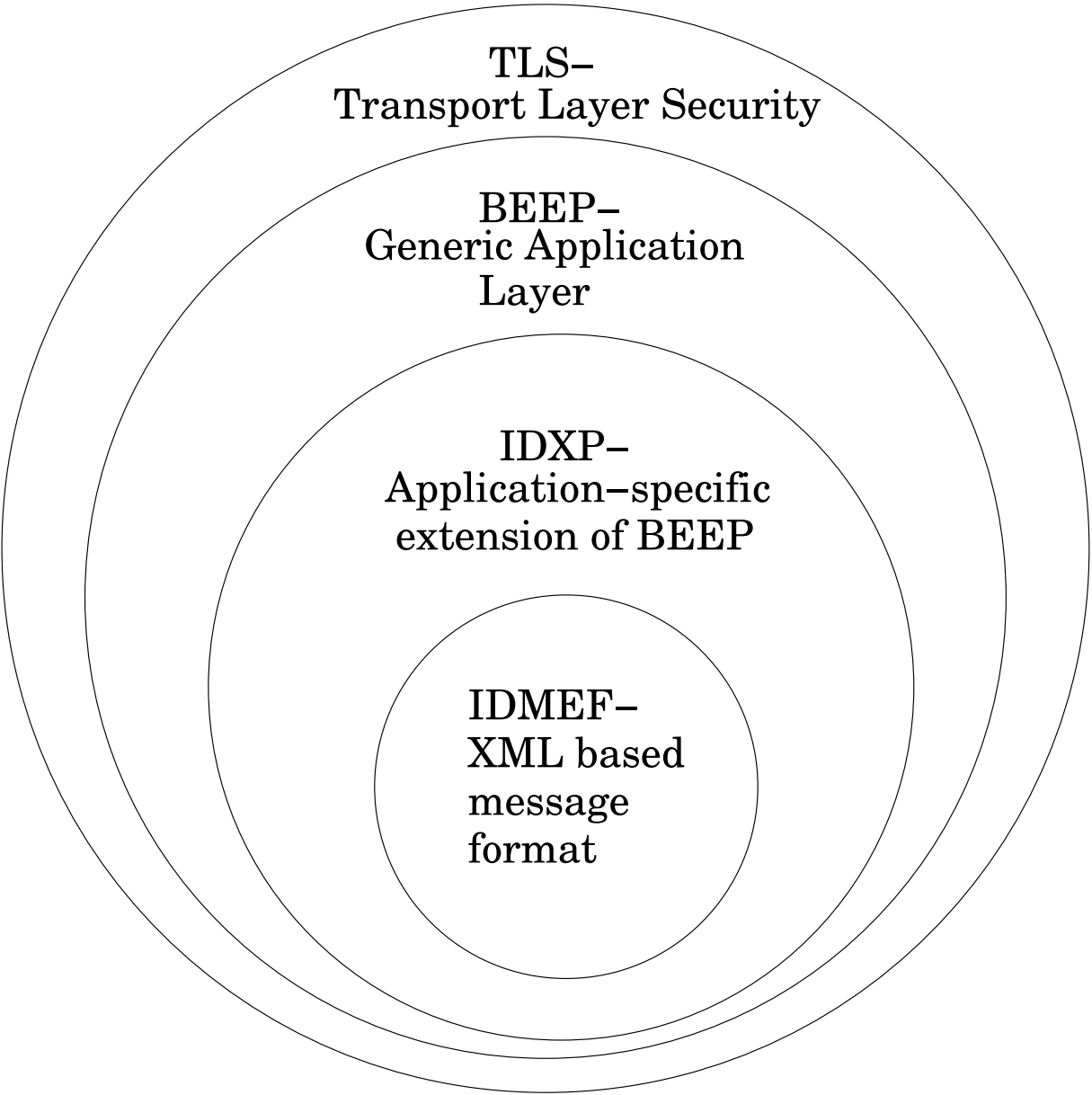
An alert using IDMEF

```
<?xml version="1.0"?>
<!DOCTYPE IDMEF-Message PUBLIC "-//IETF//DTD RFCxxxx IDMEF v0.1//EN"
"/usr/local
/share/idmef-message.dtd">
<IDMEF-Message version="0.1">
  <Alert alertid="1" impact="unknown" version="1">
    <Time>
      <ntpstamp>0x3ae0a921.0x0</ntpstamp>
      <date>2001-04-20</date>
      <time>16:24:49</time>
    </Time>
    <Analyzer ident="MORPHEUS5">
      <Node>
        <location>REC_414</location>
        <Address category="ipv4-addr">
          <address>128.10.251.104</address>
        </Address>
      </Node>
  <Alert alertid="2" impact="unknown" version="1">
    <Time>
      <ntpstamp>0x3ae448d7.0x0</ntpstamp>
      <date>2001-04-23</date>
      <time>10:23:03</time>
    </Time>
    <Analyzer ident="MORPHEUS5">
      <Node>
        <location>REC_414</location>
        <Address category="ipv4-addr">
          <address>128.10.251.104</address>
        </Address>
      </Node>
    </Analyzer>
    <Classification>
      <name>IDS152 - PING BSD</name>
      <url>No URL available</url>
    </Classification>
    <Source spoofed="unknown">
      <Node>
        <Address category="ipv4-addr">
          <address>128.10.243.21</address>
        </Address>
      </Node>
    </Source>
    <Target decoy="unknown">
      <Node>
        <Address category="ipv4-addr">
          <address>128.10.251.104</address>
        </Address>
      </Node>
    </Target>
  </Alert>
</IDMEF-Message>
```

For more information:

http://www.ietf.org/html.charters/idwg-charter.html
http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-03.txt
http://www.ietf.org/internet-drafts/draft-ietf-idwg-requirements-05.txt

IDXP, BEEP, IDMEF, XML, TLS
Sorting out the Alphabet Soup



IDMEF Daemon

The Intrusion Detection Working Group (IDWG) of the IETF is developing a protocol for the exchange of alerts between ID systems.

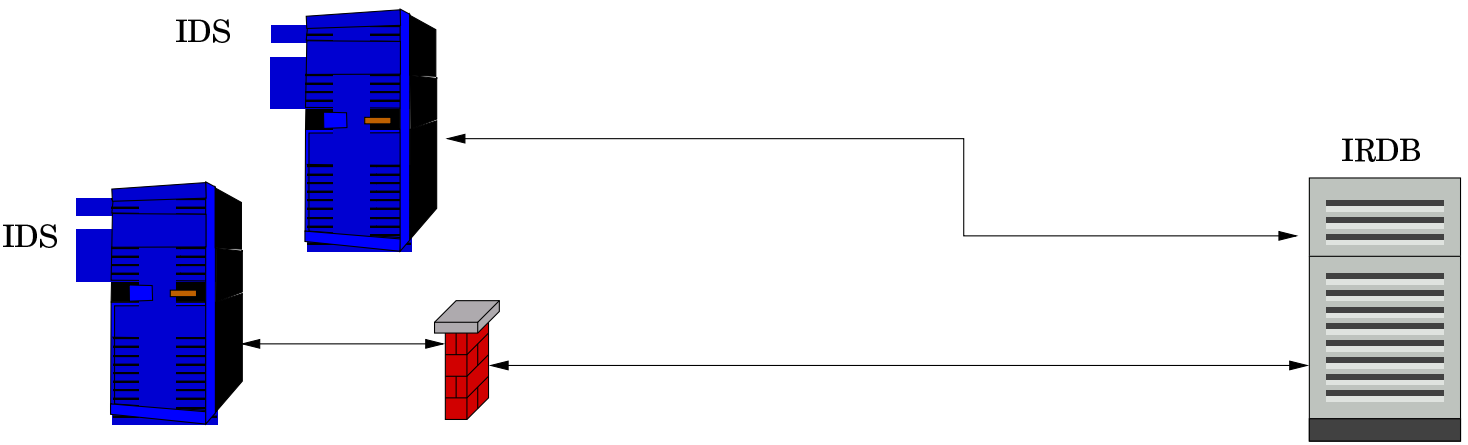
The Intrusion Detection Message Exchange Format (IDMEF) is an XML document type which provides a format for the messages. The IDWG has also developed two application layer network protocols for the exchage of IDMEF formatted data. These protocols are the Intrusion Alert Protocol (IAP) and the Intrusion Detection Exchange Protocol (IDXP). The IDWG has not officially recommended either protocol at this time, although IDXP is expected to be recommended at the next IETF meeting.

One of the goals of the IRDB project is to produce a secure user–space daemon which will receive IDMEF formatted alerts via the IDWG recommended protocol and store the alert information into the IRDB. Once stored in the database, alert information can be correlated with security incidents. It is intended that this daemon will integrate easily with the existing IDS systems as they come to support the IDMEF format and the IDWG recommended protocol.

Intrusion Alert Protocol (IAP)

The Intrusion Alert Protocol is an application layer protocol designed for the transfer of IDMEF formatted messages between Intrusion Detection Systems. It provides for encryption via TLS (SSL v3), and is partially modeled after HTTP.

IAP is similar enough to HTTP to allow it to be seamlessly passed through a standard HTTP proxy. This gives it the benefit of not requiring additional holes through a corporate firewall.



More information is available at the IDWG home page.
http://www.ietf.org/html.charters/idwg–charter.html

Intrusion Detection Exchange Protocol (IDXP)

The Intrusion Detection Exchange Protocol is an application layer protocol for the exchange of IDMEF formatted data between Intrusion Detection Systems. It is based on the Blocks Extensible Exchange Protocol (BEEP) as defined by RFC 3080.

BEEP is an attempt to create a "one–size–fits–all" application layer in order to reduce duplicated effort in the creation of future application layer protocols. It integrates features of well–known protocols such as HTTP, SMTP, and FTP while eliminating needless complexity. It is heavily based on XML and includes support for SASL authentication and TLS encryption.

BEEP uses profiles, which are analogous to XML document type descriptors, to define new application layer protocols based on the BEEP core.

The draft IDXP profile is available from the IDWG home page.
http://www.ietf.org/html.charters/idwg–charter.html