Abstract

The purpose of the Incident Object Description Exchange Format (IODEF) is to define a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. The IODEF satisfies the requirements specified in RFCXXX [1]

This Internet-Draft describes a data model for representing incident
information exported from incident handling systems managed by CSIRTs. An implementation of the data model in the Extensible Markup Language (XML) is presented, an XML Document Type Definition is developed, and examples are provided.

Table of Contents

1. Introduction ................................................. 4
   1.1 Terminology ............................................ 4
   1.2 Overview .............................................. 4
   1.3 About the IODEF Data Model ......................... 4
   1.4 About the IODEF Implementation .................. 5
   1.5 About the Transport Protocol .................. 6
   1.6 Related Work ....................................... 6

2. Formatting Issues ........................................... 8
   2.1 IODEF XML Documents ................................... 8
      2.1.1 The Document Prolog .............................. 8
      2.1.2 White Space Processing ......................... 9
      2.1.3 Languages in the IODEF ....................... 9
   2.2 IODEF Data Types ...................................... 9
      2.2.1 Integers ......................................... 10
      2.2.2 Real Numbers ................................... 10
      2.2.3 Characters and Strings ....................... 10
      2.2.4 Bytes ........................................... 10
      2.2.5 Enumerated Types ................................ 10
      2.2.6 Date-Time Strings ................................ 11
      2.2.7 Port Lists ...................................... 11
      2.2.8 Postal Address .................................. 11
      2.2.9 Person or Organization .......................... 11
      2.2.10 Telephone and Fax Numbers .................... 11
      2.2.11 Email string ................................... 11
      2.2.12 Uniform Resource Identifier strings .......... 11
      2.2.13 Timezone string ................................ 12
      2.2.14 Unique Identifiers .............................. 12

3. The IODEF Data Model .................................... 13
   3.1 IODEF-Document class ................................ 13
   3.2 Incident class ....................................... 13
   3.3 IncidentID class .................................... 16
   3.4 AlternativeID class .................................. 17
   3.5 RelatedActivity class ............................... 18
   3.6 AdditionalData ...................................... 18
3.7 Contact class ............................ 20
  3.7.1 RegistryHandle class ................. 22
3.8 Time classes ............................. 23
  3.8.1 StartTime .............................. 23
  3.8.2 EndTime ............................... 23
  3.8.3 DetectTime .................. 23
  3.8.4 ReportTime ...................... 23
3.9 Expectation class ..................... 23
3.10 Method class ........................... 25
  3.10.1 Classification class ............... 26
3.11 Assessment class ....................... 27
  3.11.1 Impact class ...................... 28
  3.11.2 TimeImpact class .................. 28
  3.11.3 MonetaryImpact class .............. 29
  3.11.4 Confidence class ................. 30
3.12 History class .......................... 30
  3.12.1 HistoryItem class ................. 31
3.13 EventData class ....................... 33
  3.13.1 Relating the Incident and EventData classes 34
  3.13.2 Cardinality of EventData .......... 35
3.14 Flow class ............................ 36
3.15 System class ........................... 36
3.16 Node class ............................. 38
  3.16.1 Counter class ....................... 40
  3.16.2 Address ............................ 40
  3.16.3 NodeRole class ..................... 42
3.17 Process class .......................... 43
3.18 Service class .......................... 43
3.19 Record class ........................... 44
  3.19.1 RecordData class .................. 45
  3.19.2 Analyzer class .................... 46
  3.19.3 RecordItem class ................. 46
4. Extending the IODEF ...................... 48
  4.1 Extending the data model .............. 48
  4.2 Extending the XML DTD ................. 48
5. Processing Considerations .............. 51
6. Internationalization issues .......... 52
7. Examples ................................. 53
  7.1 Code Red detection notification ...... 53
1. Introduction

1.1 Terminology

The key words "MUST," "MUST NOT," "REQUIRED," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," "RECOMMENDED," "MAY," and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [5].

Definitions for some of the common computer security-related terminology used in this document can be found in Section 2 of [1].

1.2 Overview

The Incident Object Description Exchange Format (IODEF) is a format for representing computer security information exchanged between Computer Security Incident Response Teams (CSIRTs). It provides a transport representation conforming to the requirements specified in [1], Requirements for Format for Incident Report Exchange.

The overriding purpose of the IODEF is to expand and enhance the operational capabilities of CSIRTs. Community adoption of the IODEF provides an improved ability to resolve incidents by simplifying collaboration and data sharing. This structured format provided by the IODEF allows for:

- increased automation in processing of incident data, since the
resources of security analysts to parse free-form textual
documents will be reduced;

- decreased effort in normalizing similar data (even when highly
  structured) from different sources; and

- a common format on which to build interoperable tools for incident
  handling, such as correlation systems that process data from
  different sites.

Terminology, notation, and conventions of the data model and XML DTD
are presented in Sections 2. The data model is described in Section
3, and the implementation considerations are covered in Sections 4
through 6. Section 7 provides several examples of IODEF documents.
Section 8 formally specifies the XML DTD implementation of the data
model. Sections 9 and 10 address the security and IANA
considerations, respectively.

1.3 About the IODEF Data Model

The IODEF data model is a data representation that provides a
framework for sharing information commonly exchanged by CSIRTs about


Internet-Draft IODEF Data Model and Implementation November 2004

computer security incidents. A number of considerations were made in
the design of the data model.

- The intent of the data model is to support the automated
  processing of incident data. Hence, little consideration was made
  to ensure human-readability. Despite the still prevalent practice
  of manual incident report generation, this model is sufficiently
  complex that it will be unwieldy to create and process without
  software.

- The data model serves as a transport format. Therefore, its
  specific representation is not the optimal representation for
  on-disk storage, long-term archiving, or in-memory processing.

- Since there is no precise, widely agreed upon definition for an
  incident, the data model does not attempt to dictate one through
  its implementation. Rather, a broad understanding is assumed that
  is flexible enough to encompass most of the CSIRT community.
Describing an incident for all definitions would require an incredibly complex data model. Therefore, the IODEF data model only intends to be a framework to convey commonly exchanged incident information. However, it ensures that there are ample mechanisms for extensibility to support organization-specific information, and techniques to reference information kept outside of the explicit data model.

Incidents have a life-cycle that dictates the exact type, quantity, and detail of the data that will be present at a given time (e.g., newly reported incidents may only contain the most rudimentary details, but closed incidents may contain a detailed analysis). The data model deals with this situation.

Communication and coordination are central to the role of a CSIRT. Hence, tracking the source of all data is central to handling the incident. Therefore, the data model provides ways to explicitly bind information to a source, and accommodates differences in the types of parties involved in the incident (e.g., varying levels of confidence in information, different data sharing arrangements).

1.4 About the IODEF Implementation

The IODEF implementation uses the Extensible Markup Language (XML) [2], specifies an XML Document Type Definition (DTD), and registers an application-specific namespace [3].

For clarity in this document, the terms "XML" and "XML documents" will be used when referring to the Extensible Markup Language (XML). The terms "IODEF description", "IODEF markup" and "IODEF document" will be used to refer to specific elements (tags) and attributes of the IODEF DTD. Finally, the terms "class" and "subclass" will be used as synonyms for an XML element.

The choice to implement the IODEF in XML was made because it provides:

- all the necessary features to define and extend a specific markup language for describing security incidents;
a well understood technique for supporting internationalization and localization;

a base of related technologies such as XSL [4], XPATH, and XML-SIG that the aid in the manipulation and use of the incident data; and

a broad community of developers who already understand how to build systems around data exchanged in this format.

While XML provides a useful implementation language for IODEF, this implementation also dictates several limitations.

XML is a text representation making it inherently inefficient either when binary data must be embedded, or very large volumes of data must be exchanged.

The data model is designed as a transport representation, and the use of XML further reinforces the inefficiency of using the IODEF for other purposes. Due to the overhead of the parser, XML is not an optimal in-memory representation. Furthermore, storing, searching, and retrieving native XML documents is problematic on a large scale dictating that this format is also a poor choice as a storage and archive format.

1.5 About the Transport Protocol

Currently, there is no transport protocol specified for exchanging IODEF documents. The working group has realized that this omission is an impediment to interoperability, and is working on identifying candidate protocols. It is likely that SOAP will be used as the messaging envelope, and HTTP will be the underlying transport.

1.6 Related Work

The IODEF is only one of several security relevant data representations being standardized. Specifically, the complementary nature of the Intrusion Detection Message Exchange Format [7] bears mention given that many incidents represented in the IODEF may have first been discovered through the use of intrusion detection system
output formatted according to the IDMEF. Given this relationship, the IODEF data model makes use of certain classes defined in the IDMEF data model.
2. Formatting Issues

2.1 IODEF XML Documents

This document uses three notations: the Unified Modeling Language (UML) to describe the data model, an XML Document Type Definition (DTD) to define the IODEF syntax, and IODEF XML markup conforming to the specified DTD to represent the incident data.

This section describes the XML notations and conventions used in this document and explains particular issues related to using them to describe the IODEF data model and syntax. For readers unfamiliar with these notations [17] and [7] will provide a comprehensive reference.

2.1.1 The Document Prolog

The "prolog" of an XML document, that part that precedes anything else, consists of the XML declaration and the document type declaration.

2.1.1.1 XML Declaration

Every IODEF document MUST begin with an XML declaration, and MUST specify the XML version used. If UTF-8 encoded is not used, the character encoding MUST also be explicitly specified.

The XML declaration with no character encoding will read as follows:

```xml
<?xml version="1.0" ?>
```

When a character encoding is specified, the XML declaration will read like the following:

```xml
<?xml version="1.0" encoding="charset" ?>
```

where "charset" is the name of the character encoding as registered with the Internet Assigned Numbers Authority (IANA), see [9].

Consistent with the XML standard, if no encoding is specified for an IODEF document, UTF-8 is assumed. IODEF documents encoded in UTF-16 MUST begin with the Byte Order Mark described by ISO/IEC 10646 Annex E and Unicode Appendix B (the "ZERO WIDTH NO-BREAK SPACE" character, #xFEFF).

2.1.1.2 IODEF DTD Formal Public Identifier
Definition described in this document is:

"-//IETF//DTD RFCxxxx IODEF v0.0//EN"

NOTE: The "RFCxxxx" text in the FPI value will be replaced with the actual RFC number when this document is published as an RFC.

This FPI MUST be used in the document type declaration within an XML document referencing the IODEF DTD as shown in Section 2.1.1.3.

2.1.1.3 IODEF DTD Document Type Declaration

The document type declaration for an XML document referencing the IODEF DTD MUST be specified either by referencing the FPI (see Section 2.1.1.2) as follows:

<!DOCTYPE IODEF-Document PUBLIC "-//IETF//DTD RFCxxxx IODEF v0.0//EN">

or by providing a URI that references a copy of the DTD as follows:

<!DOCTYPE IODEF-Document SYSTEM "/path/to/IODEF-Document.dtd">

2.1.2 White Space Processing

All IODEF elements support the "xml:space" attribute. If "xml:space" is set to "preserve," the IODEF processing application MUST treat all white space in the element's content as significant. If "xml:space" is set to "default," the application is free to decide on the handling of the whitespace.

2.1.3 Languages in the IODEF

For the IODEF elements that support free-form text, the "xml:lang" attribute can be used to identify the language of its contents. The valid language codes for the "xml:lang" attribute are described in RFC 3066 [6].

IODEF messages SHOULD specify the language in which their contents are encoded. In general, the language can be specified with the
"xml:lang" attribute in the top-level element and letting all other elements "inherit" that definition.

If no language is specified, English SHOULD be assumed.

2.2 IODEF Data Types

The IODEF data model defines a number of data types.

2.2.1 Integers

Integer attributes are represented by the INTEGER data type. Integer data MUST be encoded in Base 10 or Base 16.

Base 10 integer encoding uses the digits '0' through '9' and an optional sign ('+' or '-'). For example, "123", "-456".

Base 16 integer encoding uses the digits '0' through '9' and 'a' through 'f' (or their upper case equivalents), and is preceded by the characters "0x". For example, "0x1a2b".

2.2.2 Real Numbers

Real (floating-point) attributes are represented by the REAL data type. Real data MUST be encoded in Base 10.

Real encoding is that of the POSIX "strtod" library function: an optional sign ('+' or '-') followed by a non-empty string of decimal digits, optionally containing a radix character, then an optional exponent part. An exponent part consists of an 'e' or 'E', followed by an optional sign, followed by one or more decimal digits. For example, "123.45e02", "-567,89e-03".

IODEF-compliant applications MUST support both the '.' and ',' radix characters.

2.2.3 Characters and Strings

Single-character attributes are represented by the CHARACTER data type. Multi-character attributes of known length are represented by the STRING data type.
Character and string data have no special formatting requirements, other than the need to occasionally use character references to represent special characters.

2.2.4 Bytes

Binary data is represented by the BYTE (and BYTE[]) data type.

Binary data MUST be encoded in its entirety using character code references (see ).

2.2.5 Enumerated Types

Enumerated types are represented by the ENUM data type, and consist of an ordered list of acceptable values. Each value has a representative keyword. Within an IODEF DTD, the enumerated type keywords are used as attribute values.

2.2.6 Date-Time Strings

Date-time strings are represented by the DATETIME data type. Each date-time string identifies a particular instant in time; ranges are not supported.

Date-time strings are formatted according to a subset of ISO 8601:2000 [13] documented in RFC 3339 [12].

2.2.7 Port Lists

A list of network ports are represented by the PORTLIST data type, and consist of a comma-separated list of numbers (individual integers) and ranges (N-M means ports N through M, inclusive). Any combination of numbers and ranges may be used in a single list. For example, "5-25,37,42,43,53,69-119,123-514".

2.2.8 Postal Address

A postal address is represented by the POSTAL data type. The format of this address data is documented in Sections 5.17 - 5.19 of RFC 2256 [10].
2.2.9 Person or Organization

The name of an individual or organization is represented by the NAME data type. The format of the NAME data type is documented in Section 5.4 of RFC 2256 [10].

2.2.10 Telephone and Fax Numbers

A telephone number is represented by the PHONE data type. The format of the PHONE data type is documented in Section 5.21 of RFC 2256 [10].

2.2.11 Email string

An email address is represented by the EMAIL data type. The format of the EMAIL data type is documented in Section 3.4.1 RFC 2822 [11].

2.2.12 Uniform Resource Identifier strings

A uniform resource identifier (URI) is represented by the URI data type. The format of the URI data type is documented in RFC 2396 [8].

2.2.13 Timezone string

A timezone is represented by the TIMEZONE data type. Its format is yet to be specified.

2.2.14 Unique Identifiers

A unique identifier in the context of particular creator of IODEF documents (e.g., a CSIRT) is represented by the UID data type. A globally unique identifier is represented by the GUID data type. The UID and GUID data types are constructed from alphanumeric strings.
3. The IODEF Data Model

In this section, the individual components of the IODEF data model will be discussed in detail. For each class, the semantics will be documented and the relationship with other classes will be depicted with UML.

3.1 IODEF-Document class

The IODEF-Document class is the top level class in the IODEF data model. All IODEF documents are an instance of this class.
Figure 1: IODEF-Document class

The aggregate class that constitutes IODEF-Document is:

Incident
  One or more. The information related to a single incident.

The IODEF-Document class has one attribute:

version
  Required. STRING. The IODEF specification version number to which the IODEF document conforms. The value of this attribute MUST be 1.0

3.2 Incident class

Every incident is represented by an instance of the Incident class. This class provides a standardized representation for commonly exchanged incident data and associates a CSIRT assigned unique identifier with the described activity.
Figure 2: the Incident class

The aggregate classes that constitute Incident are:

IncidentID
One. An incident tracking number assigned to this incident by the CSIRT that generated the IODEF document.

AlternativeID
Zero or one. A list of incident tracking numbers used by other CSIRTs to refer to the incident described in the document.

RelatedActivity
Zero or one. A list of incident tracking numbers of related incidents.

Description
Zero or more. STRING. A free-form textual description of the incident.

Assessment
One or more. A characterization of the impact of the incident.

Method
Zero or more. The techniques used by the intruder in the incident.
DetectTime
   Zero or one. The time the incident was first detected.

StartTime
   Zero or one. The time the incident started.

EndTime
   Zero or one. The time the incident ended.

ReportTime
   One. The time the incident was reported.

Contact
   One or more. Contact information for the parties involved in the incident.

Expectation
   Zero or more. Expected action to be performed by the recipient of the document.

History
   Zero or one. A log of significant events or actions that occurred during the course of handling the incident.

EventData
   Zero or more. Description of the events comprising the incident.

AdditionalData
   Zero or more. Mechanism by which to extend the data model.

The Incident class has two attributes:

purpose
   Required. ENUM. The purpose attribute represents the reason why the IODEF document was created. It is closely related to the Expectation class (Section 3.9). This attribute is defined as an enumerated list:

   1. handling. The document was sent for incident-handling purposes;

   2. statistics. The document was sent to be included in a data-repository for statistical purposes;

   3. warning. The document was sent as a warning;
4. other. The document was sent for purposes specified in the Expectation element.

The value of this attribute is logically inherited by the children of this class. That is to say, the disclosure rules applied to this class, also apply to its children.

It is possible to set a granular disclosure policy, since all of the high-level classes (i.e., children of the Incident class) have a restriction attribute. Therefore, a child can override the guidelines of a parent class, be it to restrict or relax the disclosure rules (i.e., a child has a weaker policy than an ancestor; or an ancestor has a weak policy, and the children selectively apply more rigid controls). The implicit value of the restriction attribute for a class that did not specify one can be found in the closest ancestor that did specify a value.

This attribute is defined as an enumerated value with a default value of "private".

Note that the default value of the restriction attribute is only defined in the context of the Incident class. In other classes where this attribute is used, no default is specified.

1. public. There are no restrictions placed in the information;
2. need-to-know. The information may be shared with other parties that are involved in the incident (e.g., multiple victim sites can be informed of each other);
3. private. The information may not be shared.
4. default. The information can be shared according to an information disclosure policy pre-arranged by the communicating parties.
3.3 IncidentID class

The IncidentID class represents an incident tracking number (UID) that is unique in the context of the CSIRT and identifies the activity characterized in an IODEF-Document.

![Diagram of the IncidentID class]

Figure 3: the IncidentID class

The IncidentID class has one attribute:

name

Required. GUID. An identifier for the CSIRT that created the IODEF-Document.

3.4 AlternativeID class

The AlternativeID class lists the incident tracking numbers used by other CSIRTs to refer to activity described in this IODEF document. Thus, a tracking number listed as an AlternativeID references the same incident detected by another CSIRT. The incident tracking numbers of the CSIRT that generated the IODEF document should never be considered an AlternativeID.

![Diagram of the AlternativeID class]

| ENUM restriction |<>--{1..*}--[ IncidentID ] |
The aggregate class that constitutes AlternativeID is:

**IncidentID**

One or more. The incident tracking number of another CSIRT.

The AlternativeID class has one attribute:

**restriction**

Optional. ENUM. This attribute has been defined in Section 3.2.

---

3.5 RelatedActivity class

The RelatedActivity class lists the incident tracking numbers of incidents that are related to the one described in the IODEF document. These references may be to local incident tracking numbers, as well as, to those of other CSIRTs.

The specifics of how a CSIRT came to believe that two incidents are related is considered out of scope.

```
+------------------+
| RelatedActivity  |
+------------------+
| ENUM restriction |<>--{1..*}--[ IncidentID ]
|                  |
+------------------+
```

Figure 5: RelatedActivity class

The aggregate class that constitutes RelatedActivity is:

**IncidentID**

One or more. The incident tracking number of a related incident.
The RelatedActivity class has one attribute:

restriction
  Optional. ENUM. This attribute has been defined in Section 3.2.

3.6 AdditionalData

The AdditionalData class serves as an extension mechanism for information not otherwise represented in the data model. For relatively simple information, atomic data types (e.g., integers, strings) are provided with a mechanism to annotate their meaning. The class can also be used to extend the data model and the DTD to support proprietary extensions by encapsulating entire XML documents conforming to another DTD (e.g., IDMEF). A detailed discussion for extending the data model and the DTD can be found in Section 4.

Unlike XML, which is self-describing, atomic data must be documented to convey its meaning. This information is described in the 'meaning' attribute. Since these description are outside the scope of the specification, some additional coordination may be required to ensure that a recipient of a document using the AdditionalData classes can make sense of the custom extensions.

The AdditionalData class has three attributes:

restriction
  Optional. ENUM. This attribute has been defined in Section 3.2.

type
Required. ENUM. The data type of the element content. The permitted values for this attribute are shown below. The default value is "string".

1. boolean. The element contains a boolean value, i.e., the strings "true" or "false"

2. byte. The element content is a single 8-bit byte (see Section 2.2.4);

3. character. The element content is a single character (see Section 2.2.3);

4. date-time. The element content is a date-time string (see Section 2.2.6);

5. integer. The element content is an integer (see Section 2.2.1);

6. portlist. The element content is a port list (see Section 2.2.7);

7. real. The element content is a real number (see Section 2.2.2);

8. string. The element content is a string (see Section 2.2.3);

9. xml. The element content is XML-tagged data (see Section 4).

Optional. STRING. A description of the semantics of the custom data in this class.

3.7 Contact class

The Contact class describes contact information for organizations and personnel involved in the incident. This class allows for the naming of the involved party, specifying contact information for them, and
identifying their role in the incident.

People and organizations are treated interchangeably as contacts; one can be associated with the other using the recursive definition of the class (the Contact class is aggregated into the Contact class). The 'type' attribute disambiguates the type of contact information being provided.

This recursive definition provides a way to relate information without requiring the explicit use of identifiers in the classes. For example, separate contact information for two individuals from the same organization would not require duplicating the organization information.

```
+------------------+
<table>
<thead>
<tr>
<th>Contact</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENUM restriction</td>
</tr>
<tr>
<td>ENUM role</td>
</tr>
<tr>
<td>ENUM type</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
+------------------+
```

Figure 7: the Contact class

The aggregate classes that constitute the Contact class are:

- **name**
  - Zero or one. NAME. The name of the contact. The contact may either be an organization or a person. The type attribute disambiguates the semantics.

- **Description**
  - Zero or one. STRING. Free-form description of the this contact. In the case of a person, this is often the organizational title of
the individual.

RegistryHandle
Zero or many. A handle name in a registry.

PostalAddress
Zero or one. POSTAL. The postal address of the contact formatted according to Section 2.2.8.

Email
Zero or many. EMAIL. The email address of the contact formatted according to Section 2.2.11.

Telephone
Zero or many. PHONE. The telephone number of the contact formatted according to Section 2.2.10.

Fax
Zero or one. PHONE. The facsimile telephone number of the contact formatted according to Section 2.2.10.

Timezone
Zero or one. TIMEZONE. The timezone in which the contact resides formatted according to Section 2.2.13.

Contact
Zero or many. Recursive definition of Contact allowing for the grouping of information.

The Contact class has three attributes:

restriction
Optional. ENUM. This attribute is defined in Section 3.2.

role
Required. ENUM. Indicates the role the contact fulfills. This attribute is defined as an enumerated list:

1. creator. The entity that generate the IODEF document.
2. admin. An administrative contact for a host or network.
3. tech. A technical contact for a host or network.
4. irt. The CSIRT involved in handling the incident.
5. cc. An entity that is to be kept informed about the handling of the incident.

**type**
Required. ENUM. Indicates the type of contact being described. This attribute is defined as an enumerated list:

1. person.
2. organization.

### 3.7.1 RegistryHandle class

The RegistryHandle class represents a handle to an Internet registry or community-specific database. A handle consists of a name specified in the element content, and the database to which it belongs specified in the type attribute.

```
+------------------+
| RegistryHandle   |
+------------------+
| STRING           |
|                  |
| ENUM type        |
+------------------+
```

**Figure 8: The RegistryHandle class**

The RegistryHandle class has one attribute:

**type**
Required. ENUM. The database to which the handle belongs. The default value is 'local'. The possible values are:

1. internic. Internet Network Information Center
2. apnic. Asia Pacific Network Information Center
3. arin. American Registry for Internet Numbers
4. lacnic. Regional Latin-American and Caribbean IP Address Registry
5. ripe. Reseaux IP Europeens
6. local. A database local to the CSIRT.

3.8 Time classes

The data model uses five different classes to represent a timestamp. Their definition is identical, but each has a distinct name to convey a difference in semantics.

The element content of each class is a timestamp formatted according to the DATETIME data type (see Section 2.2.6).

```
+----------------------------------+
| {Start| End| Report| Detect}Time |
+----------------------------------+
| DATETIME                          |
+----------------------------------+
```

Figure 9: the Time classes

3.8.1 StartTime

The StartTime class represents the time the incident began.

3.8.2 EndTime

The EndTime class represents the time the incident ended.

3.8.3 DetectTime

The DetectTime class represents the time the first activity of the incident was detected.

3.8.4 ReportTime

The ReportTime class represents the time the incident was reported. This timestamp SHOULD coincide to the time at which the IODEF document is generated.

3.8.5 DateTime
The DateTime class is a generic representation of a timestamp. Its semantics should be inferred from the parent class into which it is aggregated.

3.9 Expectation class

The Expectation class conveys to the recipient of the IODEF document the actions the sender is requesting.

The aggregate classes that constitute Expectation are:

Description
One or many. STRING. A free-form description of the desired action(s).

StartTime
Zero or one. The time at which the action should be performed. A timestamp that is earlier than the ReportTime specified in the Incident class denotes that the expectation should be fulfilled as soon as possible. The absence of this element leaves the execution of the expectation to the discretion of the recipient.

EndTime
Zero or one. The time by which the action should be completed. If the action is not carried out by this time, it should no longer be performed.

Contact
Zero or one. The expected actor for the action.

The Expectations class has three attributes:
restriction  
Optional. ENUM. This attribute is defined in Section 3.2.

priority  
Optional. ENUM. Indicates the desired priority of the action. This attribute is an enumerated list with no default value.

1. low. Low priority
2. medium. Medium priority
3. high. High priority

category  
Optional. ENUM. Classifies the type of action requested. This attribute is an enumerated list with no default value.

1. nothing. No action is requested. Do nothing with the information.
2. contact-site. Contact the listed site in the recipient's constituency.
3. contact-me. Contact the originator of the document.
4. investigate. Investigate the machine(s) listed in the document.
5. block. Block traffic from the machine(s) listed in the document.
6. other. Perform some custom action described in the Description class.

3.10 Method class  
The Method class describes the methodology used by the intruder to
perpetrate the events of the incident. This class can reference well-known vulnerability or exploit databases; the intruder tools used in the attack; and provide a free-form description of the activity.

```
+------------------+
| Method           |
+------------------+
| ENUM restriction |<>--{0..*}--[ Classification ] |
|                  |
|                  |<>--{0..*}--[ Description ]    |
+------------------+
```

Figure 11: The Method class

The Method class is composed of two aggregate classes.

Classification
Zero or many. A reference to a well-known vulnerability or exploit databases.

Description
Zero or many. STRING. A free-form text description of the methodology used by the intruder.

The Method class has one attribute:

restriction
Optional. ENUM. This attribute is defined in Section 3.2.

3.10.1 Classification class

The Classification class is a reference to an external database of computer vulnerabilities, exposures, or viruses. A reference consists of the database name, the entry name in the database, and the URI to this entry.

+------------------+
The aggregate classes that constitute Classification:

name
One.  STRING.  The key into the database specified in the origin attribute.

url
Zero or One.  URI.  A URL to additional information about the vulnerability or exposure referenced by the name.

The Classification class has two attribute:

restriction
Optional.  ENUM.  This attribute is defined in Section 3.2.

origin
Required.  ENUM.  The name of the database to which the reference is being made. The permitted values are shown below.

1.  bugtraqid.  Bugtraq
2.  cve.  Mitre Common Vulnerabilities or Exposures
3.  certcc.  CERT Coordination Center Vulnerability Catalog
4.  vendor.  A product vendor whose name should be specified in the name class
5.  local.  A local database.
6.  other.  A custom database whose URL is specified in the url class, and the name of the entry is specified in the name class.
### 3.11 Assessment class

The Assessment class describes the technical and non-technical repercussions of the incident on the CSIRT's constituency.

Note: The IODEF definition of the Assessment class reuses the IDMEF definition (see Section 4.2.4.5 of [7]), but also extends it.

```
+------------------+
| Assessment       |
+------------------+
| ENUM restriction |<>--{0..*}--[ Impact ]  |
|                  |<>--{0..*}--[ TimeImpact ] |
|                  |<>--{0..*}--[ MonetaryImpact ] |
|                  |<>--{0..1}--[ Confidence ]  |
+------------------+
```

**Figure 13: Assessment class**

The aggregate classes that constitute Assessment are:

- **Impact**
  - Zero or many. Technical impact of the incident on a network.

- **TimeImpact**
  - Zero or many. Impact of the activity measured with respect to time.

- **MonetaryImpact**
  - Zero or many. Impact of the activity measured with respect to financial loss.

- **Confidence**
  - Zero or one. An estimate of confidence in the assessment.

The Assessment class has one attribute:

- **restriction**
  - Optional. ENUM. This attribute is defined in Section 3.2.
3.11.1 Impact class

The Impact class allows for categorizing and describing the technical impact of the incident on the network of an organization.

Note: The IODEF definition of the Impact class reuses the IDMEF definition (see Section 4.2.6.1 of [7]).

3.11.2 TimeImpact class

The TimeImpact class describes the impact of the incident on an organization as a function of time. It provides a way to convey down time and recovery time.

```
+------------------+
| TimeImpact       |
+------------------+
| REAL             |
| ENUM severity    |
| ENUM metric      |
| ENUM units       |
+------------------+
```

Figure 14: TimeImpact class

The element content will be a numeric value (REAL) specifying a unit of time. The unit and metric attributes will imply the semantics of the element content.

The TimeImpact class has three attributes:

severity
Optional. ENUM. An estimate of the relative severity of the activity. The permitted values are shown below. There is no default value.

1. low. Low severity
2. medium. Medium severity
3. high. High severity

**metric**
Required. ENUM. Defines the metric in which the time is expressed. The permitted values are shown below. There is no default value.

1. labor. Total staff-time to recovery from the activity (e.g., 2 employees working 4 hours each would be 8 hours)

2. elapsed. Elapsed time from the beginning of the recovery to its completion.

3. downtime. Duration of time for which some provided service(s) was not available.

**units**
Required. ENUM. Defines the units in which the element content is expressed. The permitted values are shown below. The default value is "hours".

1. seconds. Seconds.

2. minutes. Minutes.

3. hours. Hours.

4. days. Days.

### 3.11.3 MonetaryImpact class

The MonetaryImpact class describes the financial impact of the activity on an organization. For example, this impact may consider losses due to the cost of the investigation or recovery, diminished productivity of the staff, or a tarnished reputation that will affect future opportunities.
The element content will be a numeric value (REAL) specifying a unit of currency described in the currency attribute.

The MonetaryImpact class has two attributes:

**severity**
Optional. ENUM. An estimate of the relative severity of the activity. The permitted values are shown below. There is no default value.

1. low. Low severity
2. medium. Medium severity
3. high. High severity

**currency**
Required. ENUM. Defines the currency in which the monetary impact is expressed. The permitted values are defined in ISO 4217:2001, Codes for the representation of currencies and funds [16]. There is no default value.

### 3.11.4 Confidence class

The Confidence class represents a best estimate of the validity and accuracy of the described impact (see Section 3.11) of the incident activity. This estimate can be expressed as a category, or a numeric
calculation.

Note: The IODEF definition of the Confidence class reuses the IDMEF definition (see Section 4.2.6.3 of [7]).

3.12 History class

The History class is a log of the significant events or actions performed by the involved parties during the course of handling the incident.

The level of detail maintained in this log is left up to the discretion of those handling the incident.

```
+------------------+
| History          |
+------------------+
| ENUM restriction |<>--{1..*}--[ HistoryItem ]
|                  |
+------------------+
```

Figure 16: The History class

The class that constitutes History is:

**HistoryItem**

One or many. Entry in the history log of significant events or actions performed by the involved parties.

The History class has one attribute:

**restriction**

Optional. ENUM. This attribute is defined in Section 3.2.

3.12.1 HistoryItem class

The HistoryItem class is an entry in the History (Section 3.12) log that documents a particular action or event that occurred in the course of handling the incident. The details of the entry are a
free-form description, but each can be categorized with the type attribute.

The aggregate classes that constitute HistoryItem are:

**IncidentID**
Zero or One. In a history log created by multiple parties, the IncidentID provides a mechanism to specify which CSIRT created a particular entry and references this organization's incident tracking number. When a single organization is maintaining the log, this class can be ignored.

**DateTime**
One. Timestamp of the this entry in the history log (e.g., when the action described in the Description was taken).

**Description**
One or many. STRING. A free-form textual description of the
action or event.

The HistoryItem class has two attributes:

restriction
Optional. ENUM. This attribute has been defined in Section 3.2.

type
Optional. ENUM. Classifies the type of activity or event documented in this history log entry. The possible values are an enumerated list whose default value is "other":

1. triaged. The incident data was received and processed by an IHS.

2. notification. Notification to an involved party in the incident was sent (e.g., a CSIRT sending a message to the attacking site).

3. shared-info. Information about this incident was shared with party not directly involved.

4. received-info. Additional information about the incident was received.

5. remediation. The incident has been resolved; a short description may be included.

6. other. A custom entry.

3.13 EventData class

The EventData class describes the events of the incident surrounding a particular set of hosts or networks. This description includes the systems from which the activity originated and those targeted, an assessment of the techniques used by the intruder, the impact of the activity on the organization, and any forensic evidence discovered.
Figure 18: The EventData class

The aggregate classes that constitute EventData are:

Description  
Zero or more. STRING. A free-form textual description of the event.

DetectTime  
Zero or one. The time the event was detected.

StartTime  
Zero or one. The time the event started.

EndTime  
Zero or one. The time the event ended.

Contact  
Zero or more. The different parties involved in the incident.

Assessment  
Zero or one. The impact of the incident on the target and the actions taken.

Method
The EventData class has one attribute:

restriction
  Optional. ENUM. This attribute is defined in Section 3.2.

3.13.1 Relating the Incident and EventData classes

There is substantial overlap in the Incident and EventData classes. Nevertheless, the semantics of these classes are quite different. The Incident class provides summary information about the entire incident, while the EventData class provides information about the individual events comprising the incident. In the most common case, the EventData class will provide more specific information for the general description provided in the Incident class. However, it may also be possible that the overall summarized information about the incident conflicts with some individual information in an EventData class when there is a substantial composition of various events in the an incident.

3.13.2 Cardinality of EventData

The EventData class can be thought of as a container for the
properties of an event in an incident. These properties include: the hosts involved, impact of the incident activity on the hosts, forensic logs, etc. With an instance of the EventData class, hosts (i.e., System class) are grouped around these common properties.

The recursive definition of the EventData class (the EventData class is aggregated into the EventData class) provides a way to related information without requiring the explicit use of unique attribute identifiers in the classes or duplicating information. Instead, the relative depth (nesting) of a class is used to group (relate) information.

Nested EventData classes imply that while the child classes share the properties of the parent, there is some properties for which they do not agree. Therefore, in order express these distinct properties, the nesting approach was used. In such a scheme, a parent EventData class MUST always have more than one EventData child.

For example, an EventData class might be used to describe two machines involved in an incident. This description can be achieved using multiple instances of the System class. It happens that there is a common technical contact (i.e., Contact class) for these two machines, but the impact (i.e., Assessment class) on them is different. A depiction of the representation for this situation can be found in Figure 19.

```
+------------------+
| EventData        |
+------------------+
    |<>----[ Contact ]|
    |<>----[ EventData ]<>----[ System ]
    |
    |<>----[ EventData ]<>----[ System ]
    |
    +------------------+
```

Figure 19: Recursion in the EventData class
3.14 Flow class

The Flow class groups the source and target hosts or networks in an event.

```
+------------------+
| Flow             |
+------------------+
    |<>--{1..*}--[ System ]
+------------------+
```

Figure 20: the Flow class

The aggregate class that constitutes Flow is:

```
System
One or More. A host or network involved in the incident activity.
```

The Flow System class has no attributes.

3.15 System class

The System class represents a computer or network involved in the incident.

The systems represented by this class are categorized according to the role they played in the incident through the category attribute. The value of this category attribute dictates the semantics of the aggregated classes in the System class. If the category attribute has a value of 'source', then the aggregated classes denote the machine and service from which the activity is originating. With a category attribute value of 'target' or 'intermediary', then the machine or service is the one targeted in the activity.
Figure 21: the System class

The aggregate classes that constitute System are:

Node
   One. A host or network involved in the incident.

Service
   Zero or more. A network service running on the system.

Counter
   Zero or more. A counter with which to summarizes properties of this host or network.

The System class has four attribute:

restriction
   Optional. ENUM. This attribute is defined in Section 3.2.

category
   Required. ENUM. Classifies the role the host or network played in the incident. The possible values are:

   1. source. The System was the source of the attack.

   2. target. The System was the target of the attack.

   3. intermediate. The System was an intermediary in the attack.

interface
   Optional. STRING. Specifies the interface on which the event(s)
on this System originated. If the Node class specifies a network rather than a host, this attribute has no meaning.

**spoofed**

Optional. ENUM. An indication of confidence as to whether this System was the true target or attacking host. The permitted values for this attribute are shown below. The default value is "unknown".

1. unknown. The accuracy of the category attribute value is unknown

2. yes. The category attribute value is probably incorrect. In the case of a source, the System is likely a decoy; with a target, the System was likely not the intended victim.

3. no. The category attribute value is believed to be correct.

# 3.16 Node class

The Node class identifies a host, network device, or network.

The base definition of the class is reused from the IDMEF specification, see Section 4.2.7.1 of [7]. However, the class has been extended by adding the NodeRole and DateTime classes.

+-------------------+
|       Node        |
+-------------------+
| ENUM category     |
| <-{0..1}--[ Location ] |
| <-{0..1}--[ name ] |
| <-{0..*}--[ Address ] |
| <-{0..1}--[ DateTime ] |
| <-{0..*}--[ NodeRole ] |
| <-{0..*}--[ Counter ] |

**Figure 22: The Node class**

The aggregate classes that constitute Node are:
Location
Zero or one. STRING. A free-from description of the physical location of the equipment.

name
Zero or one. STRING. The name of the equipment (e.g., fully qualified domain name). This information MUST be provided if no Address information is given.

Address
Zero or more. The hardware, network, or application address of the Node. Unless a name is provided, at least one address must be specified.

DateTime
Zero or one. A timestamp of when the resolution between the name and address was performed. This information SHOULD be provided if both an Address and name are given.

NodeRole
Zero or more. The intended purpose of the equipment.

Counter
Zero or more. A counter with which to summarizes properties of this host or network.

The Node class has one attribute:

category
Optional. ENUM. The context in which the Address and name classes should be considered, if relevant. The permitted values for this attribute are shown below. The default value is "unknown".

1. unknown. Domain unknown or not relevant
2. ads. Windows 2000 Advanced Directory Services
3. afs. Andrew File System (Transarc)
4. coda. Coda Distributed File System

5. dfs. Distributed File System (IBM)

6. dns. Domain Name System

7. hosts. Local hosts file

8. kerberos. Kerberos realm

9. nds. Novell Directory Services

10. nis. Network Information Services (Sun)

11. nisplus. Network Information Services Plus (Sun)

12. nt. Windows NT domain

13. wfw. Windows for Workgroups

3.16.1 Counter class

The Counter class summarize multiple occurrences of some event, or conveys counts on various features (e.g., packets, sessions, events).

The value of the counter is the element content, with its units represented in the type attribute. The complete semantics are entirely context dependant based on the class in which the Counter is aggregated.

+------------------+
| Counter          |
+------------------+
| INTEGER          |
| ENUM type        |
| STRING meaning   |
Figure 23: the Counter class

The Counter class has two attributes:

- **type**
  - Optional. ENUM. Specifies the units of the element contents.
  - 1. packet. Count of packets.
  - 2. session. Count of sessions
  - 3. event. Count of events
  - 4. other. User defined count

- **meaning**
  - Optional. STRING. Describes the semantics of the element content if the type attribute is set to other.

### 3.16.2 Address

The Address class represents a hardware (layer-2), network (layer-3), or application (layer-7) address.

This class was originally derived from the IDMEF specification [7].
category
   Required. ENUM. The type of address represented. The permitted values for this attribute are shown below. The default value is "ipv4-addr".

1. atm. Asynchronous Transfer Mode (ATM)
2. mac. Media Access Control (MAC) address
3. sna. IBM Shared Network Architecture (SNA) address
4. ipv4-addr. IPv4 host address in dotted-decimal notation (a.b.c.d)
5. ipv4-net. IPv4 network address in dotted-decimal notation, slash, significant bits (a.b.c.d/nn)
6. ipv4-net-mask. IPv4 network address in dotted-decimal notation, slash, network mask in dotted-decimal notation (a.b.c.d/w.x.y.z)
7. ipv6-addr. IPv6 host address
8. ipv6-net. IPv6 network address, slash, significant bits
9. ipv6-net-mask. IPv6 network address, slash, network mask
10. vm. IBM VM ("PROFS") email address
11. e-mail. Electronic mail address (RFC 822)
12. lotus-notes. Lotus Notes e-mail address

vlan-name
   Optional. STRING. The name of the Virtual LAN to which the address belongs.
vlans-num
  Optional.  STRING.  The number of the Virtual LAN to which the
  address belongs.

3.16.3  NodeRole class

The NodeRole class describes (based on a pre-defined list) the
function performed by a particular host.

+---------------+
| NodeRole      |
+---------------+
| STRING        |
|               |
| ENUM category |
+---------------+

Figure 25: The NodeRole class

The element content should be empty in all cases other than when the
category attribute is set to "other".

The NodeRole class has one attributes:

category
  Required.  Functionality provided by a node.  If a value of
  "other" is specified, a description SHOULD be provided in the
element content.  The default value is "other".

  1.  client.  Client computer
  2.  server-internal.  Server with internal services
  3.  server-public.  Server with public services
  4.  www.  WWW server
  5.  mail.  Mail server
  6.  messaging.  Messaging server (e.g.  NNTP, IRC, IM)
  7.  streaming.  Streaming-media server
8. voice. Voice server (e.g. SIP, H.323)
9. file. File server (e.g. SMB, CVS, AFS)
10. ftp. FTP server
11. p2p. Peer-to-peer node
12. name. Name server (e.g. DNS, WINS)
13. directory. Directory server (e.g. LDAP, finger, whois)
14. credential. Credential server (e.g. domain controller, Kerberos)
15. print. Print server
16. application. Application server
17. database. Database server
18. infra. Infrastructure server (e.g. router, firewall, DHCP)
19. log. Logserver
20. other. other role not in this list

3.17 Process class

The Process class describes a running program on a given host involved in an incident. This class is reused outright from the IDMEF specification, see Section 4.2.7.3 of [7].

3.18 Service class

The Service class describes a network service of a host or network. The service is identified by specific port or list of ports, along with the application listening on that port.

When Service occurs as an aggregate class of a System that is a source, then that the service is the one from which activity of interest is originating. Conversely, when Service occurs as an aggregate class of a System that is a target, then that service is the one to which activity of interest is being directed.

This class was originally derived from the IDMEF specification, see
The aggregate classes that constitute Service are:

port
Zero or one. INTEGER. A port number.

portlist
Zero or one. PORTLIST. A list of port numbers formatted according to Section 2.2.7.

Application
Zero or more. The application bound to the specified port or portlist.

The Service class must specify either a port or portlist.

The Service class has two attributes:

ip_version
Required. INTEGER. The IP version number.

ip_protocol
Required. INTEGER. The IANA protocol number.

3.19 Record class

The Record class is a container class for log and audit data that provides supportive information about the incident. The source of this data will often be the output of monitoring tools (e.g., IDMEF messages generated by an IDS, connection logs from a web server) that
were used to uncover the malicious activity. These logs should provide evidence as to why a CSIRT believes an incident has occurred.
Figure 28: The RecordData class

The aggregate classes that constitutes RecordData is:

DateTime
Zero or one. Timestamp of the RecordItem data.

Description
Zero or more. STRING. Free-form textual description of the provided RecordItem data. At minimum, this description should convey the significance of the provided RecordItem data.

Analyzer
Zero or one. Information about the sensor used to generate the RecordItem data.

RecordItem
One or more. Log, audit, or forensic data.

The RecordData class has one attributes:

restriction
Optional. ENUM. This attribute has been defined in Section 3.2.

3.19.2 Analyzer class

The Analyzer class identifies the sensor (e.g., IDS, firewall, web server) used to generate particular log or audit data. The definition of the class is reused from the IDMEF specification, see Section 4.2.7.3 of [7]. However, in this context, the definition of an analyzer is expanded beyond merely an IDS.

3.19.3 RecordItem class

The RecordItem class provides a way to incorporate relevant logs, audit trails, or forensic data to support the conclusions made during
the course of analyzing the incident. The class supports both the
direct encapsulation of the data, as well as, provides primitives to
reference data stored elsewhere.

The dtype attribute will dictate the type of log data that will be
found in this class. This class is very similar to the
AdditionalData class (Section 3.6) in that it is an extension
mechanism that can support proprietary representations of security
event data, not all of which is necessarily in XML.

+------------------+
| RecordItem       |
+------------------+
| ANY              |
|                  |
| ENUM type        |
+------------------+

Figure 29: The RecordItem class

The RecordItem class has one attribute:

**type**

Required. The type of data included in the element content. The
permitted values for this attribute are shown below. The default
value is "string".

1. boolean. The element contains a boolean value, i.e., the
strings "true" or "false"

2. byte. The element content is a single 8-bit byte (see Section 2.2.4);

3. character. The element content is a single character (see Section 2.2.3);

4. date-time. The element content is a date-time string (see Section 2.2.6);

5. integer. The element content is an integer (see Section 2.2.1);

6. portlist. The element content is a port list (see Section 2.2.7);

7. real. The element content is a real number (see Section 2.2.2);

8. string. The element content is a string (see Section 2.2.3);

9. file. The element content is a base64 encoded binary file;

10. path. The element content is a filesystem path;

11. url. The element content is a URL (see Section 2.2.12);

12. xml. The element content is XML-tagged data (see Section 4).

4. Extending the IODEF

In order to support the changing activity of CSIRTS, the IODEF data model and DTD will need to evolve along with them. To allow new features to be added, both the data model and the DTD can be extended as described in this section. As these extensions mature, they can be incorporated into future versions of the specification or published separately.

4.1 Extending the data model
There are two mechanisms for extending the IODEF data model:
inheritance and aggregation.

- By using inheritance, new subclasses may be derived and given
  additional attributes or operations not found in the superclass.

- Aggregation allows for entirely new, self-contained classes to be
  created and associated with a parent class.

Of the two extension mechanisms, inheritance is preferred, because it
preserves the existing data model and the operations (methods)
executed on the classes of the model. There are explicit guidelines
for extending the XML DTD (see Section 4.2) which set limits on where
extensions to the data model may be made.

4.2 Extending the XML DTD

There are two ways to extend the IODEF XML DTD:

1. The AdditionalData (see Section 3.6) and RecordItem (see Section
   3.19.3) classes allow implementers to include arbitrary "atomic"
data. (e.g., integers, strings). This approach SHOULD be used
whenever possible.

2. The AdditionalData and RecordItem classes also allow implementers
to extend the IODEF XML DTD with additional DTDs that describe
arbitrarily complex data types and relationships.

The following guidelines MUST be followed when extending the IODEF
DTD with another DTD in the extension classes:

1. The IODEF description MUST include a document type declaration
   (see Section 2.1.1.3);

2. The document type declaration MUST define a parameter entity that
   contains the location of the extension DTD, and then reference

```
<!DOCTYPE IODEF-Document SYSTEM "/path/to/IODEF-Document.dtd"
  [ <!ENTITY % x-extension SYSTEM "/path/to/extension.dtd">
```
In this example, the "x-extension" parameter entity is defined and then referenced, causing the DTD for the extension to be read by the XML parser. The name of the parameter entity defined for this purpose MUST be a string beginning with "x-"; there are no other restrictions on the name (other than those imposed on all entity names by XML). Multiple extensions may be included by defining multiple entities and referencing them. For example:

```xml
<!DOCTYPE IODEF-Document SYSTEM "/path/to/IODEF-Document.dtd"
[ <!ENTITY % x-extension SYSTEM "/path/to/extension.dtd">
  <!ENTITY % x-another SYSTEM "/path/to/another.dtd">
  %x-extension;
  %x-another; ]>
```

3. Extension DTDs MUST declare all of their elements and attributes in a separate XML namespace. Extension DTDs MUST NOT declare any elements or attributes in the "IODEF" or default namespaces.

For example, the "test" extension might be declared as follows:

```xml
<!ELEMENT test:test ( test:a, test:b, test:c )>
<!ATTLIST test:test
  xmlns      CDATA   #IMPLIED
  xmlns:test CDATA   #IMPLIED
>
<!ELEMENT test:a (#PCDATA)>
<!ATTLIST test:a
  test:attr   CDATA   #IMPLIED
>
<!ELEMENT test:b (#PCDATA)>
<!ELEMENT test:c (#PCDATA)>
```

4. Extensions MUST only be included in the AdditionalData or RecordItem classes whose "type" attribute is "xml". For example:
<IODEF-Document version="0.0">
  <Incident ident="...">
    ...
    <AdditionalData type="xml">
      <test:test
        xmlns:test="http://www.ietf.org/iodef/test.html"
        xmlns="http://www.ietf.org/iodef/test.html">
        <test:a test:attr="...">...</test:a>
        <test:b>...</test:b>
        <test:c>...</test:c>
      </test:test>
    </AdditionalData>
  </Incident>
</IODEF-Document>
5. Processing Considerations

The IODEF documents MUST be well-formed, and when practical, SHOULD also be valid.

It is expected that IODEF-compliant applications will normally not include the IODEF DTD in their communications. Instead, the DTD will be referenced in the document type declaration section of the IODEF document (see Section 2.1.1.3).

On occasion, an IODEF-compliant application may receive a well-formed, or well-formed and valid IODEF document containing tags or content in the tags that are not expected. These spurious conditions might include:

- Unrecognized tags used in one of the extension classes (i.e., AdditionalData or RecordItem);
- Unrecognized tags outside of the extension classes; or
- Well-formed and validate document where element or attribute values to not conform to the expected values identified by an enumerated list;

IODEF-compliant applications MUST continue to process IODEF documents that contain unknown tags, provided that these documents are well-formed. It is up to the individual application to decide how to process any content from the unknown tag.
6. Internationalization issues

Internationalization and localization is of specific concern to the IODEF, since it is only through collaboration, often across language barriers, that certain incidents be resolved. The IODEF supports this goal by depending on XML constructs, and through explicit design choices in the data model.

The IODEF leverages that XML natively supports different character encodings. This flexibility allows information encoded in an IODEF document to be in most languages. In order to disambiguate the explicit language used on a per-element basis, XML provides the xml:lang attribute. Using the xml:lang attribute allows the IODEF to make use of multiple languages in the same document.

The intent of the data model was to provide internationalization and localization, but not to the detriment of inter-operability. While IODEF does support different languages, the data model also relies heavily on standardized enumerated attributes that can crudely approximate the contents of the document. With this approach, a CSIRT should be able to make some sense of an IODEF document it might receive that uses a language unfamiliar to its analysts.

Likewise, the data model was designed so that classes where free-text might be used for descriptive purposes always have a one-to-many cardinality with its parent (i.e., Description class). The primary intent of this design was to allow the same description to be repeated in another instance of the class but in a different language. This approach allows recipients speaking different languages to receive the identical document, but allows the IODEF parser to select the appropriate language.
7. Examples

This section provides representative examples of incident data converted to an IODEF document.

7.1 Code Red detection notification

The following email message is a typical example of an incident report where one host is infected with a worm. The original report sent by email is presented in Figure 34, and the corresponding equivalent as an IODEF document is shown below.

From e-citizen@domain.com
Date: 13 Sep 2001 23:19:24 -0000
To: cert-domain@domain.com
Subject: 10.1.1.2 - Code Red Virus detected

Automated message,
you don't have to reply to this email.

Your system with the IP number 10.1.1.2 seems to be infected with the Code Red virus.

For more information see http://www.domain.org/react/code_redII.html
Please fix the problem or inform a person who is responsible for that machine to do so.

>From our web server logs (Port 80):
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

Figure 34: Code Red detection notification: initial report

<IODEF-Document version="1.0">
  <Incident restriction="need-to-know" purpose="handling">
    <IncidentID name="CERT-DOMAIN.COM">CERT-DOMAIN.COM#189</IncidentID>
    <Description>Host sending out Code Red probes</Description>
    <Assessment>
      <Impact severity="low" completion="failed" type="none"></Impact>
    </Assessment>
    <ReportTime>2001-09-13T23:19:24+00:00</ReportTime>
    <Contact role="creator" role="irt" type="organization">
      <name>CERT-FOR-OUR-DOMAIN.PL</name>
      <Email>cert-for-our-domain.pl@ourdomain.pl</Email>
    </Contact>
    <Contact role="tech" type="organization">
      <name>Constituency-contact for 10.1.1.2</name>
      <Email>Constituency-contact@10.1.1.2.pl</Email>
    </Contact>
    <Expectation category="investigate">
      <Description>Track and clean host</Description>
    </Expectation>
    <EventData>
      <Flow>
        <System category="source">
          <Node>
            <Address category="ipv4-addr">10.1.1.2</Address>
          </Node>
        </System>
      </Flow>
    </EventData>
  </Incident>
</IODEF-Document>
Figure 35: Code Red detection notification: CSIRT response

7.2 IODEF-Document with XML signature

7.3 IODEF-Document encrypted using XML encryption

7.4 IODEF-Document encrypted and signed using XML signature & encryption
8. The IODEF Document Type Definition
<?xml version="1.0" encoding="UTF-8"?>
<!--
*******************************************************************************
*******************************************************************************
*** Incident Object Description Exchange Format DTD ***
***               Version 01, November 2004                      ***
*******************************************************************************
*******************************************************************************
-->
<!ENTITY % attlist.iodef "
  version             CDATA                   #FIXED    '0.30'
"

<!--
====================================================================
==  Element definitions                                           ==
====================================================================
-->
<!--
====================================================================
== IODEF-Document class                                           ==
====================================================================
-->
<!ELEMENT IODEF-Document (Incident+)>
<!ATTLIST IODEF-Document
  %attlist.iodef;
  xmlns:iodef CDATA #FIXED "urn:iana:xml:ns:iodef"
>
<!--
====================================================================
==  Incident class                                                ==
====================================================================
-->
<!ELEMENT Incident (IncidentID, AlternativeID?, RelatedActivity?, Description*, Contact+, Rep
<!ATTLIST Incident
  restriction %attvals.restriction; "private"
  purpose %attvals.purpose; #REQUIRED
>
<!--
====================================================================
==  IncidentID class                                              ==
====================================================================
-->
<!ELEMENT IncidentID (#PCDATA)>
<!ATTLIST IncidentID
  name CDATA #IMPLIED
>
restriction %attvals.restriction; #IMPLIED
>
<!--
====================================================================
==  AlternativeID class                                           ==
====================================================================
-->
<!ELEMENT AlternativeID (IncidentID+)>  
<!ATTLIST AlternativeID
    restriction %attvals.restriction; #IMPLIED
>
<!--
====================================================================
==  RelatedActivity class                                         ==
====================================================================
-->
<!ELEMENT RelatedActivity (IncidentID+)>  
<!ATTLIST RelatedActivity
    restriction %attvals.restriction; #IMPLIED
>
<!--
====================================================================
===  AdditionalData class                                        ===
====================================================================
-->
<!ELEMENT AdditionalData ANY>  
<!ATTLIST AdditionalData
    restriction %attvals.restriction; #IMPLIED
    type %attvals.dtype; #REQUIRED
    meaning CDATA #IMPLIED
>
<!--
====================================================================
===  Contact class                                               ===
===    - Name
===    - RegistryHandle
===    - PostalAddress
===    - Email
===    - Telephone
===    - Fax
===    - Timezone
===    - Contact (recursive)
====================================================================
-->
<!ELEMENT Contact (Name?, Description*, RegistryHandle*, PostalAddress?, Email*, Telephone*, Fax*, Timezone*)>
<!ATTLIST Contact
    contactrole (creator | admin | tech | irt | cc) #REQUIRED
contacttype (person | organization) #REQUIRED

<restriction %attvals.restriction; #IMPLIED>

<!ELEMENT RegistryHandle (#PCDATA)>
<!ATTLIST RegistryHandle
type %attvals.registrytype; "local"
>
<!ELEMENT PostalAddress (#PCDATA)>
<!ATTLIST PostalAddress
lang NMTOKEN #IMPLIED>

<!ELEMENT Email (#PCDATA)>
<!ELEMENT Telephone (#PCDATA)>
<!ELEMENT Fax (#PCDATA)>

<!--
====================================================================
===  Time-based classes                                          ===
===    - DateTime
===    - ReportTime
===    - DetectTime
===    - StartTime
===    - EndTime
====================================================================
-->
<!ELEMENT DateTime (#PCDATA)>
<!ELEMENT ReportTime (#PCDATA)>
<!ELEMENT DetectTime (#PCDATA)>
<!ATTLIST DetectTime

<!ELEMENT StartTime (#PCDATA)>
<!ELEMENT EndTime (#PCDATA)>

<!--
====================================================================
===  History class                                               ===
====================================================================
-->
<!ELEMENT History (HistoryItem+)>
<!ATTLIST History

restriction %attvals.restriction; #IMPLIED
>
<!ELEMENT HistoryItem (DateTime, IncidentID?, Description+)>
<!ATTLIST HistoryItem
type %attvals.historycat; #IMPLIED
restriction %attvals.restriction; #IMPLIED
>
</!


Internet-Draft IODEF Data Model and Implementation November 2004

<!ELEMENT Expectation (Description+, Contact?, StartTime?, EndTime?)>
<!ATTLIST Expectation
priority %attvals.priority; #IMPLIED
restriction %attvals.restriction; #IMPLIED
category %attvals.expectcat; #IMPLIED
>
</!

<!ELEMENT Method (Classification*, Description*)>
<!ATTLIST Method
restriction %attvals.restriction; #IMPLIED
>
</!

<!ELEMENT Classification (name, url?)>
<!ATTLIST Classification
restriction %attvals.restriction; #IMPLIED
origin %attvals.origin; "other"
>
</!

<!ELEMENT Assessment (DateTime, IncidentID?, Description+)>
<!ATTLIST Assessment
priority %attvals.priority; #IMPLIED
restriction %attvals.restriction; #IMPLIED
category %attvals.assessmentcat; #IMPLIED
>
</!

Internet-Draft    IODEF Data Model and Implementation      November 2004

metric (days | hours | minutes | seconds) "hours"

metric (days | hours | minutes | seconds) "hours"

metric (days | hours | minutes | seconds) "hours"

metric (days | hours | minutes | seconds) "hours"

metric (days | hours | minutes | seconds) "hours"
<!--====================================================================
===  Flow and System class                                       ===
===  Note. Represents merged Source and Target classes of IDMEF     ===
===  (sections 4.2.4.3, 4.2.4.4)                                   ===
====================================================================-->
<!ELEMENT Flow (System+)>
<!ELEMENT System (Node, Service*, Counter*)>
<!ATTLIST System
category %attvals.systemcat; #IMPLIED
spoofed %attvals.spoofed; "unknown"
interface CDATA #IMPLIED
restriction %attvals.restriction; #IMPLIED
->

<!--====================================================================
====== Node class                                                ===
===      - Address
===      - NodeRole
===      - Location
===    Note. IODEF Node class is extended IDMEF Node class (4.2.7.1):
===        <!ELEMENT Node ( location?, (name | Address), Address* )>===
====================================================================-->
<!ELEMENT Node (name?, Address*, DateTime?, Location?, NodeRole*, Counter*)>
<!ATTLIST Node
category %attvals.nodecat; "unknown"
->
<!ELEMENT Address (#PCDATA)>
<!ATTLIST Address
category %attvals.addrcat; "ipv4-addr"
vlan-name CDATA #IMPLIED
vlan-num CDATA #IMPLIED
->
<!ELEMENT Location (#PCDATA)>
<!ATTLIST Location
lang NMTOKEN #IMPLIED
<!ELEMENT NodeRole (#PCDATA)>
<!ATTLIST NodeRole
  category %attvals.noderolecat; "other"
  lang NMTOKEN #IMPLIED>

<!ELEMENT Counter (#PCDATA)>
<!ATTLIST Counter
  type (packet | session | event | other )  "other"
  meaning CDATA #IMPLIED>

<!ELEMENT Service ((port | portlist), Application?)>
<!ATTLIST Service
  ip_version CDATA  "4"
  ip_protocol CDATA #REQUIRED

<!ELEMENT port (#PCDATA)>
<!ELEMENT portlist (#PCDATA)>

<!ELEMENT Application (name, url?)>
<!ATTLIST Application
  appid  CDATA   "0"
  configid CDATA "0"
  vendor_name CDATA #IMPLIED
  version CDATA #IMPLIED
<!ELEMENT Record (RecordData+)>  
<!ATTLIST Record
  restriction %attvals.restriction; #IMPLIED
>
<!ELEMENT RecordData (Description*, DateTime?, Analyzer?, RecordItem+)>
<!ATTLIST RecordData
  ident CDATA "0"
  restriction %attvals.restriction; #IMPLIED
>
<!ELEMENT Analyzer (Node?, Process?)>
<!ATTLIST Analyzer
  analyzerid CDATA "0"
  manufacturer CDATA #IMPLIED
  model CDATA #IMPLIED
  version CDATA #IMPLIED
  class CDATA #IMPLIED
  ostype CDATA #IMPLIED
  osversion CDATA #IMPLIED
>
<!ELEMENT Process (name, pid?, path?, arg*, env*)>
<!ATTLIST Process
  ident CDATA "0"
>
<!ELEMENT path (#PCDATA)>  
<!ELEMENT env (#PCDATA)>  
<!ELEMENT pid (#PCDATA)>  

<!ATTLIST RecordItem
  ...
dtye %attvals.dtye; #REQUIRED

<!--
====================================================================
=== Simple classes containing multilingual content ===
   - Description
   - Contact.Name
====================================================================
-->
<!ELEMENT Description ANY>
<!ATTLIST Description
preserve %attvals.preserve; #IMPLIED
transform %attvals.transform; #IMPLIED
lang NMTOKEN #IMPLIED
>
<!ELEMENT Name ANY>
<!ATTLIST Name
preserve %attvals.preserve; #IMPLIED
transform %attvals.transform; #IMPLIED
lang NMTOKEN #IMPLIED
>
<!--
====================================================================
=== Miscellaneous simple classes ===
   - url
   - name
====================================================================
-->
<!ELEMENT name (#PCDATA)>
<!ATTLIST name
lang NMTOKEN #IMPLIED
>
<!ELEMENT url (#PCDATA)>
<!--
====================================================================
=== Attribute list declarations. ===
====================================================================
-->
<!--
| Attributes of the IODEF element. In general, the fixed value
| of this attribute will change each time a new version of
| the DTD is released.
-->
<!--
SECTION 2. Attribute value declarations. Enumerated values for the many element-specific attribute lists.

-->  

<!--  
| Defines purpose of the Incident  
-->  

<!ENTITY % attvals.purpose "
  ( handling | statistics | warning | other )"
>

<!--  
| Defines restriction on access to an element's content  
-->  

<!ENTITY % attvals.restriction "
  ( default | public | need-to-know | private )"
>

<!--  
| Values for the Expectation.expectcat attributes  
-->  

<!ENTITY % attvals.expectcat "
  ( nothing | contact-site | contact-me | investigate | block | other )"
>

<!--  
| Values for the AdditionalData.type attribute.  
-->  

<!ENTITY % attvals.adtype "
  ( boolean | byte | character | date-time | integer | ntpstamp | portlist | real | string | xml )"
>

<!--  
| Values for the RecordItem.type attribute  
-->  

<!ENTITY % attvals.dtype "
  ( boolean | byte | character | date-time | integer | ntpstamp | portlist | real | string | file | path | url | xml )"
>

<!--  
| Values for the History.type attribute.  
-->  

<!ENTITY % attvals.historycat "
  ( triaged | notification | shared-info | received-info | remediation | other )"
>

<!--  
| Values for the Address.category attribute.  
-->

Internet-Draft    IODEF Data Model and Implementation      November 2004

<!ENTITY % attvals.addrcat "
( atm | mac | sna |
ipv4-addr | ipv4-net | ipv4-net-mask |
ipv6-addr | ipv6-net | ipv6-net-mask |
asn |
vm | e-mail | lotus-notes )
"
</!
| Values for the Id.type attribute.
-->

<!ENTITY % attvals.idtype "
( current-user | original-user | target-user | user-privils |
current-group | group-privils )
"
</!
| Values for the Impact.completion attribute.
-->

<!ENTITY % attvals.completion "
( failed | succeeded )
"
</!
| Values for the Impact.type attribute.
-->

<!ENTITY % attvals.impacttype "
( none | admin | dos | file | recon | user | unknown |
other )
"
</!
| Values for the RegistryHandle.type attribute.
-->

<!ENTITY % attvals.registrytype "
( internic | apnic | arin | lacnic | ripencc | local )
"
</!
| Values for the Confidence.rating attribute.
-->

<!ENTITY % attvals.rating "
( low | medium | high | numeric | unknown )
"
</!

--->
Values for the Impact.severity attribute.

```
<!ENTITY % attvals.severity "
   ( low | medium | high )
">
```

Values for the Node.category attribute.

```
<!ENTITY % attvals.nodecat "
   ( unknown | ads | afs | coda | dfs | dns | hosts |
     kerberos | nds | nis | nisplus | nt | wfw )
">
```

Values for the NodeRole.category attribute.

```
<!ENTITY % attvals.noderolecat "
   ( client | server-internal | server-public | www | mail |
     messaging | streaming | voice | file | ftp | p2p | name |
     directory | credential | print | application | database |
     infra | log | other )
">
```

Values for the Classification.origin attribute.

```
<!ENTITY % attvals.origin "
   ( bugtraqid | cve | certcc | vendor | local | other)
">
```

Values for the System.spoofed attribute

```
<!ENTITY % attvals.spoofed "
   ( unknown | yes | no )
">
```

Values for the System.category attribute

```
<!ENTITY % attvals.systemcat "
   ( source | target | intermediate )
"
9. Security considerations

Due to the sensitive nature of some of the data that might be represented in the IODEF, the integrity, confidentiality, and non-repudiation of these documents in transit SHOULD be ensured. Although this protection can be provided by the transport mechanism, applying this security to the IODEF document itself is RECOMMENDED.

When used, the applied protective measures MUST use cryptographic techniques. XML Digital Signatures [14] SHOULD be used for ensuring integrity and non-repudiation, while XML Encryption [15] SHOULD be used to ensure the confidentiality of an IODEF document. Examples using signatures and encryption on an IODEF document can be found in Section 7:

- IODEF-Document with XML signature (Section 7.2)
- IODEF-Document encrypted using XML encryption (Section 7.3)
- IODEF-Document encrypted and signed using XML signature & encryption (Section 7.4)

Additional information on applying XML Digital Signatures and XML Encryption to an IODEF document can be found in the IODEF Implementation Guide [18].
10. IANA considerations
11. Acknowledgments

The following groups contributed substantially to this document and should be recognized for their efforts.

- the Incident Object Description and Exchange Format Working-Group of the TERENA task-force (TF-CSIRT)

- the eCSIRT.net project
12. References

12.1 Normative References


1.0 (Second Edition)", October 2000,

<http://www.w3.org/TR/REC-xml-names/>.

(XSL) Version 1.0", October 2001,
<http://www.w3.org/TR/xsl/>.

[5] Bradner, S., "Key words for use in RFCs to Indicate Requirement


[9] Freed, N., "IANA Charset Registration Procedures", BCP 2278,


[13] International Organization for Standardization, "International
Standard: Data elements and interchange formats - Information
interchange - Representation of dates and times", ISO 8601,

12.2 Informative References


Authors' Addresses

Jan Meijer
SURFnet bv
P.O. Box 19035
Utrecht NL-3501 DA
Netherlands
Phone: +31 302 305 305
EMail: jan.meijer@surfnet.nl

Roman Danyliw
CERT Coordination Center
4500 Fifth Ave.
Pittsburgh, PA 15213
USA
Phone: +1 412 268 7090
EMail: rdd@cert.org

Yuri Demchenko
NLnet Labs
Netherlands
EMail: demch@chello.nl
Copyright (C) The Internet Society (2004). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE REPRESENTS THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.