              Incident Object Description Exchange Format Usage Guidance

Abstract

   The Incident Object Description Exchange Format (IODEF) v2 (RFC 7970)
   defines a data representation that provides a framework for sharing
   information about computer security incidents commonly exchanged by
   Computer Security Incident Response Teams (CSIRTs). Since the IODEF
   model includes a wealth of available options that can be used to
   describe a security incident or issue, it can be challenging for
   security practitioners to develop tools that leverage IODEF for
   incident sharing. This document provides guidelines for IODEF
   implementers. It addresses how common security indicators can be
   represented in IODEF and provides use cases of how IODEF is being
   used. This document aims to make IODEF's adoption by vendors easier
   and to encourage faster and wider adoption of the model by CSIRTs
   around the world.

Status of This Memo

Table of Contents

1. Introduction

   The Incident Object Description Exchange Format (IODEF) v2 [RFC7970]
   defines a data representation that provides a framework for sharing
   computer security incident information commonly exchanged by Computer
   Security Incident Response Teams (CSIRTs). The IODEF data model
   consists of multiple classes and data types that are defined in the
   IODEF XML schema.

   The IODEF schema was designed to describe all the possible fields
   needed in a security incident exchange. Thus, IODEF contains a
   plethora of data constructs that could make it hard for IODEF
   implementers to decide which are important. Additionally, in the
   IODEF schema, there exist multiple fields and classes that do not
   necessarily need to be used in every possible data exchange.
   Moreover, some IODEF classes are useful only in rare circumstances.
   This document tries to address these concerns. It also presents how
   common security indicators can be represented in IODEF, it points out
   the most important IODEF classes for an implementer and describes
   other ones that are not as important, and it presents some common
   pitfalls for IODEF implementers and how to address them. The end
   goal of this document is to make IODEF's use by vendors easier and to
   encourage wider adoption of the model by CSIRTs around the world.

   Section 3 discusses the recommended classes and how an IODEF
   implementer should choose the classes to implement. Section 4
   presents common considerations a practitioner will come across and
   how to address them. Section 5 goes over some common uses of IODEF.

2. Terminology

   The terminology used in this document is defined in [RFC7970].

3. Implementation and Use Strategy

   It is important for IODEF implementers to distinguish how the IODEF
   classes will be used in incident information exchanges. It is also
   important to understand the most common IODEF classes that describe
   common security incidents or indicators. This section describes the
   most important classes and factors an IODEF practitioner should take
   into consideration before using IODEF or designing an implementation.

3.1. Minimal IODEF Document

   An IODEF document must include at least an Incident class, an
   xml:lang attribute that defines the supported language, and the IODEF
   version attribute. An Incident must contain a purpose attribute and
   three mandatory-to-implement elements. These elements are

GenerationTime class (which describes the time of the incident), an
IncidentID class, and at least one Contact class. The structure of
the minimal IODEF-Document class is shown in Figure 1.

```
+---------------+              +--------------+
|IODEF-Document |              | Incident     |
+---------------+              +--------------+              +--------------+
|STRING version |<>--{1..*}--| ENUM purpose |<>---------|  IncidentID  |
|ENUM xml:lang  |              |              |              +--------------+
|               |              |              |              | STRING name  |
+---------------+              |              |              +--------------+
                               |              |
                               |              |
                               |              |<>---------[GenerationTime]
                               |              |
                               |              |              +--------------+
                               |              |<>-{1..*}--[ Contact       |
                               +--------------+              +--------------+
                                                             | ENUM role    |
                                                             | ENUM type    |
                                                             +--------------+
```
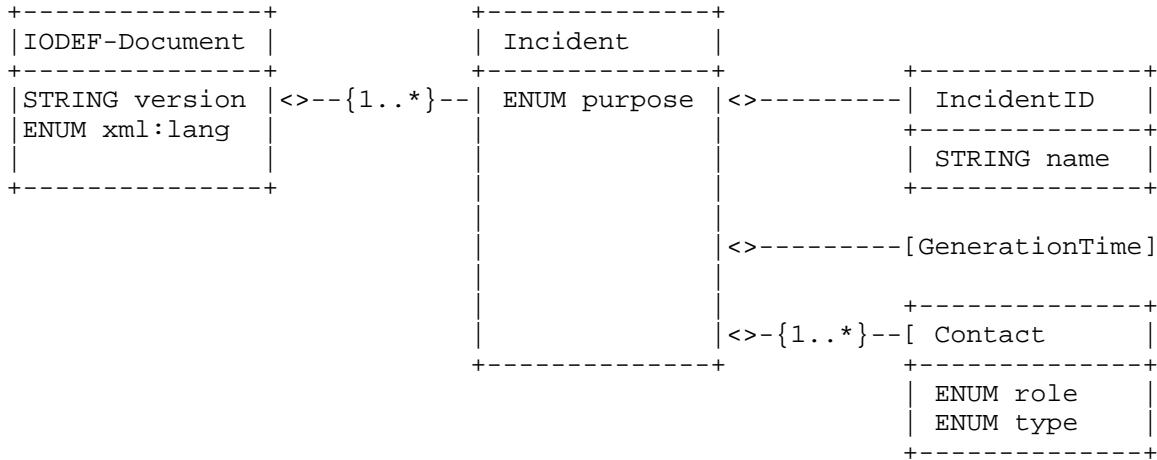
                 Figure 1: Minimal IODEF-Document Class

   The IncidentID class must contain at least a name attribute.

   In turn, the Contact class requires the type and role attributes, but
   no elements are required by the IODEF v2 specification.
   Nevertheless, at least one of the elements in the Contact class, such
   as an Email class, should be implemented so that the IODEF document
   is useful.

   Section 7.1 of [RFC7970] presents a minimal IODEF document with only
   the mandatory classes and attributes. Implementers can also refer to
   Section 7 of [RFC7970] and Appendix B of this document for examples
   of documents that are IODEF v2.

3.2. Information Represented

   There is no need for a practitioner to use or implement IODEF classes
   and fields other than the minimal ones (see Section 3.1) and the ones
   necessary for her use cases. The implementer should carefully look
   into the schema and decide which classes to implement (or not).

   For example, if we have Distributed Denial of Service (DDoS) as a
   potential use case, then the Flow class and its included information
   are the most important classes to use. The Flow class describes
   information related to the attacker and victim hosts, which could
   help automated filtering or sinkhole operations.

Another potential use case is malware command and control (C2).
After modern malware infects a device, it usually proceeds to connect
to one or more C2 servers to receive instructions from its master and
potentially exfiltrate information. To protect against such
activity, it is important to interrupt the C2 communication by
filtering the activity. IODEF can describe C2 activities using the
Flow and the ServiceName classes.

For use cases where indicators need to be described, the
IndicatorData class will be implemented instead of the EventData
class.

In summary, an implementer should identify the use cases and find the
classes that are necessary to support in IODEF v2. Implementing and
parsing all IODEF classes can be cumbersome, in some occasions, and
unnecessary. Other external schemata can also be used in IODEF to
describe incidents or indicators. External schemata should be parsed
accordingly only if the implementer's IODEF use cases require
external schema information. But even when an IODEF implementation
cannot parse an external schema, the IODEF report can still be
valuable to an incident response team. The information can also be
useful when shared further with content consumers that are able to
parse this information.

IODEF supports multiple language translations of free-form, ML_STRING
text in all classes [RFC7970]. That way, text in Description
elements can be translated to different languages by using a
translation identifier in the class. Implementers should be able to
parse iodef:MLStringType classes and extract only the information
relevant to languages of interest.

3.3. IODEF Classes

   [RFC7970] contains classes that can describe attack Methods, Events,
   Incidents, Indicators, how they were discovered, and the Assessment
   of the repercussions for the victim. It is important for IODEF users
   to know the distinction between these classes in order to decide
   which ones fulfill their use cases.

   An IndicatorData class depicts a threat indicator or observable that
   describe a threat that resulted in an attempted attack. For example,
   we could see an attack happening (described in the IndicatorData),
   but it might have been prevented and not have resulted in an incident
   or security event. On the other hand, an EventData class usually
   describes a security event and can be considered a report of
   something that took place.

Classes like Discovery, Assessment, Method, and RecoveryTime are used
in conjunction with EventData as they relate to the incident report
described in the EventData. The RelatedActivity class can reference
an incident, an indicator, or other related threat activity.

While deciding what classes are important for the needed use cases,
IODEF users should carefully evaluate the necessary classes and how
these are used in order to avoid unnecessary work. For example, if
we want to only describe indicators in IODEF, the implementation of
Method or Assessment might not be important.

4. IODEF Usage Considerations

   Implementers need to consider some common, standardized options for
   their IODEF use strategy.

4.1. External References

   The IODEF format includes the Reference class used for externally
   defined information, such as a vulnerability, Intrusion Detection
   System (IDS) alert, malware sample, advisory, or attack technique.
   To facilitate the exchange of information, the Reference class was
   extended to the enumeration reference format [RFC7495]. The
   enumeration reference format specifies a means to use external
   enumeration specifications (e.g., Common Vulnerabilities and
   Exposures (CVE)) that could define an enumeration format, specific
   enumeration values, or both. As external enumerations can vary
   greatly, implementers should only support the ones expected to
   describe their specific use cases.

4.2. Extensions

   The IODEF data model [RFC7970] is extensible. Many attributes with
   enumerated values can be extended using the "ext-*" prefix.
   Additional classes can also be defined by using the AdditionalData
   and RecordItem classes. An extension to the AdditionalData class for
   reporting phishing emails is defined in [RFC5901]. Information about
   extending IODEF class attributes and enumerated values can be found
   in Section 5 of [RFC7970].

   Additionally, IODEF can import existing schemata by using an
   extension framework defined in [RFC7203]. The framework enables
   IODEF users to embed XML data inside an IODEF document using external
   schemata or structures defined by external specifications. Examples
   include CVE, Common Vulnerability Reporting Framework (CVRF), and
   Open Vulnerability and Assessment Language (OVAL). [RFC7203]
   enhances the IODEF capabilities without further extending the data
   model.

IODEF implementers should not use their own IODEF extensions unless
data cannot be represented using existing standards or unless
importing them in an IODEF document as defined in [RFC7203] is not a
suitable option.

4.3. Indicator Predicate Logic

An IODEF document [RFC7970] can describe incident reports and
indicators. The Indicator class can include references to other
indicators, observables, and more classes that contain details about
the indicator. When describing security indicators, it is often
common to need to group them together in order to form a group of
indicators that constitute a security threat. For example, a botnet
might have multiple command and control servers. For that reason,
IODEF v2 introduced the IndicatorExpression class, which is used to
add the indicator predicate logic when grouping more than one
indicator or observable.

Implementations must be able to parse and apply the Boolean logic
offered by an IndicatorExpression in order to evaluate the existence
of an indicator. As explained in Section 3.29.5 of [RFC7970], the
IndicatorExpression element operator defines the operator applied to
all the child elements of the IndicatorExpression. If no operator is
defined, "and" should be assumed. IndicatorExpressions can also be
nested together. Child IndicatorExpressions should be treated as
child elements of their parent, and they should be evaluated first
before being evaluated with the operator of their parent.

Users can refer to Appendix A for example uses of the
IndicatorExpressions in an IODEF v2.

4.4. Disclosure Level

Access to information in IODEF documents should be tightly locked
since the content may be confidential. IODEF has a common attribute,
called "restriction", which indicates the disclosure guideline to
which the sender expects the recipient to adhere to for the
information represented in the class and its children. That way, the
sender can express the level of disclosure for each component of an
IODEF document. Appropriate external measures could be implemented
based on the restriction level. One example is when Real-time Inter-
network Defense (RID) [RFC6545] is used to transfer the IODEF
documents, it can provide policy guidelines for handling IODEF
documents by using the RIDPolicy class.

The enforcement of the disclosure guidelines is out of scope for
IODEF. The recipient of the IODEF document needs to follow the
guidelines, but these guidelines themselves do not provide any

enforcement measures. For that purpose, implementers should consider
appropriate privacy control measures, technical or operational, for
their implementation.

5. IODEF Uses

   IODEF is currently used by various organizations in order to
   represent security incidents and share incident and threat
   information between security operations organizations.

5.1. Implementations

   In order to use IODEF, tools like IODEF parsers are necessary.
   [RFC8134] describes a set of IODEF implementations and uses by
   various vendors and Computer Emergency Readiness Team (CERT)
   organizations. The document does not specify any particular
   mandatory-to-implement (MTI) IODEF classes but provides a list of
   real-world uses. Perl and Python modules (XML::IODEF, Iodef::Pb,
   iodeflib) are some examples. Moreover, implementers are encouraged
   to refer to Section 7 of [RFC8134] for practical IODEF usage
   guidelines. On the other hand, [IODEF_IMP] includes various vendor
   incident reporting products that can consume and export in IODEF
   format.

5.2. Inter-vendor and Service Provider Exercise

   As an interoperability exercise, a limited number of vendors
   organized and executed exchanges of threat indicators in IODEF in
   2013. The transport protocol used was RID. The threat information
   shared included indicators from DDoS attacks as well as malware
   incidents and spear phishing that targets specific individuals after
   harvesting information about them. The results served as proof-of-
   concept (PoC) about how seemingly competing entities could use IODEF
   to exchange sanitized security information. As this was a PoC
   exercise, only example information (no real threats) was shared as
   part of the exchanges.

```
 _____                                    _____
| Vendor X  |                                  | Vendor Y  |
| RID Agent |_____ ------------ _____     | RID Agent |
|_____|        | Internet   |            |_____|
                      ------------
```

```
        ---- RID Report message --->
        -- carrying IODEF example ->
        --------- over TLS -------->

        <----- RID Ack message -----
        <--- in case of failure ----
```
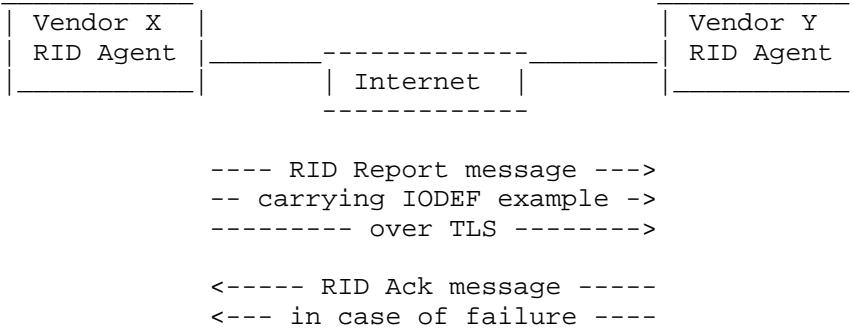
              Figure 2: PoC Peering Topology

Figure 2 shows how RID interactions took place during the PoC.
Participating organizations were running RID Agent software on
premises. The RID Agents formed peering relationships with other
participating organizations. When Entity X had a new incident to
exchange, it would package it in IODEF and send it to Entity Y over
Transport Layer Security (TLS) in a RID Report message. In case
there was an issue with the message, Entity Y would send a RID
Acknowledgement message back to Entity X, which included an
application-level message to describe the issue. Interoperability
between RID Agents implementing [RFC6545] and [RFC6546] was also
confirmed.

The first use case included sharing of malware data related to an
Incident between CSIRTs. After Entity X detected an incident, Entity
X would put data about malware found during the incident in a backend
system. Entity X then decided to share the incident information with
Entity Y about the malware discovered. This could be a human
decision or part of an automated process.

Below are the steps followed for the malware information exchange
that was taking place:

(1) Entity X has a sharing agreement with Entity Y and has already
     been configured with the IP address of Entity Y's RID Agent.

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and
     mutual authentication occurs using PKI digital certificates.

(3) Entity X pushes out a RID Report message, which contains
     information about N pieces of discovered malware. IODEF is used
     in RID to describe the

     (a) hash of malware files;

     (b) registry settings changed by the malware; and

     (c) C2 information for the malware.

(4) Entity Y receives a RID Report message and sends a RID
     Acknowledgement message.

(5) Entity Y stores the data in a format that makes it possible for
     the backend to know which source the data came from.

Another use case was sharing a DDoS attack as explained in the
following scenario: Entity X, a Critical Infrastructure and Key
Resource (CIKR) company, detects that their internet connection is
saturated with an abnormal amount of traffic. Further investigation
determines that this is an actual DDoS attack. Entity X's CSIRT
contacts their ISP, Entity Y, and shares information with them about
the attack traffic characteristics. Entity X's ISP is being
overwhelmed by the amount of traffic, so it shares attack signatures
and IP addresses of the most prolific hosts with its adjacent ISPs.

Below are the steps followed for a DDoS information exchange:

(1) Entity X has a sharing agreement with Entity Y and has already
     been configured with the IP address of Entity Y's RID Agent.

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and
     mutual authentication occurs using PKI digital certificates.

(3) Entity X pushes out a RID Report message, which contains
     information about the DDoS attack. IODEF is used in RID to
     describe the following:

     (a) Start and Detect dates and times;

     (b) IP addresses of nodes sending DDoS traffic;

     (c) sharing and use restrictions;

     (d) traffic characteristics (protocols and ports);

(e) HTTP user agents used; and

(f) IP addresses of C2 for a botnet.

(4) Entity Y receives a RID Report message and sends a RID
     Acknowledgement message.

(5) Entity Y stores the data in a format that makes it possible for
     the backend to know which source the data came from.

(6) Entity Y shares information with other ISP entities it has an
     established relationship with.

One more use case was sharing spear-phishing email information as
explained in the following scenario: the board members of several
defense contractors receive a targeted email inviting them to attend
a conference in San Francisco. The board members are asked to
provide their personally identifiable information such as their home
address, phone number, corporate email, etc., in an attached document
that came with the email. The board members are also asked to click
on a URL that would allow them to reach the sign-up page for the
conference. One of the recipients believes the email to be a
phishing attempt and forwards the email to their corporate CSIRT for
analysis. The CSIRT identifies the email as an attempted spear-
phishing incident and distributes the indicators to their sharing
partners.

Below are the steps followed for a spear-phishing information
exchange between CSIRTs that were part of this PoC.

(1) Entity X has a sharing agreement with Entity Y and has already
     been configured with the IP address of Entity Y's RID Agent.

(2) Entity X's RID Agent connects to Entity Y's RID Agent, and
     mutual authentication occurs using PKI digital certificates.

(3) Entity X pushes out a RID Report message that contains
     information about the spear-phishing email. IODEF is used in
     RID to describe the following:

     (a) attachment details (file Name, hash, size, malware family);

     (b) target description (IP, domain, NSLookup);

     (c) email information (From, Subject, header information, date/
          time, digital signature); and

     (d) confidence score.

   (4) Entity Y receives a RID Report message and sends a RID
        Acknowledgement message.

   (5) Entity Y stores the data in a format that makes it possible for
        the backend to know which source the data came from.

   Appendix B includes some of the IODEF example information that was
   exchanged by the organizations' RID Agents as part of this PoC.

5.3. Use Cases

   Other use cases of IODEF, aside from the ones described above, could
   be as follows:

   (1) ISP notifying a national CERT or organization when it identifies
        and acts upon an incident, and CERTs notifying ISPs when they
        are aware of incidents.

   (2) Suspected phishing emails could be shared amongst organizations
        and national agencies. Automation could validate web content
        that the suspicious emails are pointing to. Identified
        malicious content linked in a phishing email could then be
        shared using IODEF. Phishing campaigns could thus be subverted
        much faster by automating information sharing using IODEF.

   (3) When finding a certificate that should be revoked, a third party
        would forward an automated IODEF message to the Certification
        Authority (CA) with the full context of the certificate, and the
        CA could act accordingly after checking its validity.
        Alternatively, in the event of a compromise of the private key
        of a certificate, a third party could alert the certificate
        owner about the compromise using IODEF.

6. IANA Considerations

   This memo does not require any IANA actions.

7. Security Considerations

   This document does not incur any new security issues, because it only
   talks about the usage of IODEFv2 defined in RFC 7970. Nevertheless,
   readers of this document should refer to the Security Considerations
   section of [RFC7970].

8. References

8.1. Normative References

   [RFC5901] Cain, P. and D. Jevans, "Extensions to the IODEF-Document
              Class for Reporting Phishing", RFC 5901,
              DOI 10.17487/RFC5901, July 2010,
              <https://www.rfc-editor.org/info/rfc5901>.

   [RFC6545] Moriarty, K., "Real-time Inter-network Defense (RID)",
              RFC 6545, DOI 10.17487/RFC6545, April 2012,
              <https://www.rfc-editor.org/info/rfc6545>.

   [RFC7203] Takahashi, T., Landfield, K., and Y. Kadobayashi, "An
              Incident Object Description Exchange Format (IODEF)
              Extension for Structured Cybersecurity Information",
              RFC 7203, DOI 10.17487/RFC7203, April 2014,
              <https://www.rfc-editor.org/info/rfc7203>.

   [RFC7495] Montville, A. and D. Black, "Enumeration Reference Format
              for the Incident Object Description Exchange Format
              (IODEF)", RFC 7495, DOI 10.17487/RFC7495, March 2015,
              <https://www.rfc-editor.org/info/rfc7495>.

   [RFC7970] Danyliw, R., "The Incident Object Description Exchange
              Format Version 2", RFC 7970, DOI 10.17487/RFC7970,
              November 2016, <https://www.rfc-editor.org/info/rfc7970>.

8.2. Informative References

   [IODEF_IMP]
              "Implementations on Incident Object Description Exchange
              Format", <http://siis.realmv6.org/implementations/>.

   [RFC6546] Trammell, B., "Transport of Real-time Inter-network
              Defense (RID) Messages over HTTP/TLS", RFC 6546,
              DOI 10.17487/RFC6546, April 2012,
              <https://www.rfc-editor.org/info/rfc6546>.

   [RFC8134] Inacio, C. and D. Miyamoto, "Management Incident
              Lightweight Exchange (MILE) Implementation Report",
              RFC 8134, DOI 10.17487/RFC8134, May 2017,
              <https://www.rfc-editor.org/info/rfc8134>.

Appendix A. Indicator Predicate Logic Examples

   In the following example, the EventData class evaluates as a Flow of
   one System with source address 192.0.2.104 OR 192.0.2.106 AND target
   address 198.51.100.1.

```
<!-- ...XML code omitted... -->
 <IndicatorData>
   <Indicator>
     <IndicatorID name="csirt.example.com" version="1">
     G90823490
     </IndicatorID>
     <Description>C2 domains</Description>
     <IndicatorExpression operator="and">
       <IndicatorExpression operator="or">
         <Observable>
           <System category="source" spoofed="no">
             <Node>
               <Address category="ipv4-addr">
                 192.0.2.104
               </Address>
             </Node>
           </System>
         </Observable>
         <Observable>
           <System category="source" spoofed="no">
             <Node>
               <Address category="ipv4-addr">
                 192.0.2.106
               </Address>
             </Node>
           </System>
         </Observable>
       </IndicatorExpression>
       <Observable>
         <System category="target" spoofed="no">
           <Node>
             <Address category="ipv4-addr">
               198.51.100.1
             </Address>
           </Node>
         </System>
       </Observable>
     </IndicatorExpression>
   </Indicator>
 </IndicatorData>
<!-- ...XML code omitted... -->
```

   Similarly, the FileData Class can be an observable in an
   IndicatorExpression. The hash values of two files can be used to
   match against an indicator using Boolean "or" logic. In the
   following example, the indicator consists of either of the two files
   with two different hashes.

```
   <!-- ...XML code omitted... -->
    <IndicatorData>
      <Indicator>
        <IndicatorID name="csirt.example.com" version="1">
        A4399IWQ
        </IndicatorID>
        <Description>File hash watchlist</Description>
        <IndicatorExpression operator="or">
            <Observable>
              <FileData>
                <File>
                  <FileName>dummy.txt</FileName>
                  <HashData scope="file-contents">
                    <Hash>
                     <ds:DigestMethod Algorithm=
                     "http://www.w3.org/2001/04/xmlenc#sha256"/>
                     <ds:DigestValue>
                      141accec23e7e5157de60853cb1e01bc38042d
                      08f9086040815300b7fe75c184
                     </ds:DigestValue>
                    </Hash>
                  </HashData>
                </File>
              </FileData>
            </Observable>
            <Observable>
              <FileData>
                <File>
                  <FileName>dummy2.txt</FileName>
                  <HashData scope="file-contents">
                    <Hash>
                     <ds:DigestMethod Algorithm=
                     "http://www.w3.org/2001/04/xmlenc#sha256"/>
                     <ds:DigestValue>
                      141accec23e7e5157de60853cb1e01bc38042d
                      08f9086040815300b7fe75c184
                     </ds:DigestValue>
                    </Hash>
                  </HashData>
                </File>
              </FileData>
            </Observable>
```

```
        </IndicatorExpression>
      </Indicator>
   </IndicatorData>
   <!-- ...XML code omitted... -->
```

Appendix B. Inter-vendor and Service Provider Exercise Examples

   Below, some of the IODEF example information that was exchanged by
   the vendors as part of this proof-of-concept, inter-vendor and
   service provider exercise.

B.1. Malware Delivery URL

   This example indicates malware and a related URL for file delivery.

```
  <?xml version="1.0" encoding="UTF-8"?>
  <IODEF-Document version="2.00"
                  xmlns="urn:ietf:params:xml:ns:iodef-2.0"
                  xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <iodef:Incident purpose="reporting">
      <iodef:IncidentID name="csirt.example.com">
        189801
      </iodef:IncidentID>
      <iodef:ReportTime>2012-12-05T12:20:00+00:00</iodef:ReportTime>
      <iodef:GenerationTime>2012-12-05T12:20:00+00:00
      </iodef:GenerationTime>
      <iodef:Description>Malware and related indicators
      </iodef:Description>
      <iodef:Assessment occurrence="potential">
        <iodef:SystemImpact severity="medium" type="breach-privacy">
          <iodef:Description>Malware with C2
          </iodef:Description>
        </iodef:SystemImpact>
      </iodef:Assessment>
      <iodef:Contact role="creator" type="organization">
        <iodef:ContactName>example.com CSIRT
        </iodef:ContactName>
        <iodef:Email>
          <iodef:EmailTo>contact@csirt.example.com
          </iodef:EmailTo>
        </iodef:Email>
      </iodef:Contact>
      <iodef:EventData>
        <iodef:Flow>
          <iodef:System category="source">
            <iodef:Node>
              <iodef:Address category="ipv4-addr">192.0.2.200
```

```
                  </iodef:Address>
                        <iodef:Address category="site-uri">
                  /log-bin/lunch_install.php?aff_id=1&amp;lunch_id=1&amp;
                  maddr=&amp;action=install
                  </iodef:Address>
              </iodef:Node>
              <iodef:NodeRole category="www"/>
            </iodef:System>
          </iodef:Flow>
        </iodef:EventData>
      </iodef:Incident>
  </IODEF-Document>
```

B.2. DDoS

   The DDoS test exchanged information that described a DDoS, including
   protocols and ports, bad IP addresses, and HTTP user agent fields.
   The IODEF version used for the data representation was based on
   [RFC7970].

```
 <?xml version="1.0" encoding="UTF-8"?>
 <IODEF-Document version="2.00"
                 xmlns="urn:ietf:params:xml:ns:iodef-2.0"
                 xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <iodef:Incident purpose="reporting" restriction="default">
     <iodef:IncidentID name="csirt.example.com">
       189701
     </iodef:IncidentID>
     <iodef:DetectTime>2013-02-05T01:15:45+00:00</iodef:DetectTime>
     <iodef:StartTime>2013-02-05T00:34:45+00:00</iodef:StartTime>
     <iodef:ReportTime>2013-02-05T01:34:45+00:00</iodef:ReportTime>
     <iodef:GenerationTime>2013-02-05T01:15:45+00:00
     </iodef:GenerationTime>
     <iodef:Description>DDoS Traffic Seen</iodef:Description>
     <iodef:Assessment occurrence="actual">
       <iodef:SystemImpact severity="medium" type="availability-system">
         <iodef:Description>DDoS Traffic
         </iodef:Description>
       </iodef:SystemImpact>
       <iodef:Confidence rating="high"/>
     </iodef:Assessment>
     <iodef:Contact role="creator" type="organization">
       <iodef:ContactName>Dummy Test</iodef:ContactName>
       <iodef:Email>
         <iodef:EmailTo>contact@dummytest.com
         </iodef:EmailTo>
       </iodef:Email>
```

```
      </iodef:Contact>
      <iodef:EventData>
        <iodef:Description>
          Dummy Test sharing with ISP1
        </iodef:Description>
        <iodef:Method>
          <iodef:Reference>
            <iodef:URL>
              http://blog.spiderlabs.com/2011/01/loic-ddos-
              analysis-and-detection.html
            </iodef:URL>
            <iodef:URL>
              http://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon
            </iodef:URL>
            <iodef:Description>
              Low Orbit Ion Cannon User Agent
            </iodef:Description>
          </iodef:Reference>
        </iodef:Method>
        <iodef:Flow>
          <iodef:System category="source" spoofed="no">
            <iodef:Node>
              <iodef:Address category="ipv4-addr">
                192.0.2.104
              </iodef:Address>
            </iodef:Node>
            <iodef:Service ip-protocol="6">
              <iodef:Port>1337</iodef:Port>
            </iodef:Service>
          </iodef:System>
          <iodef:System category="source" spoofed="no">
            <iodef:Node>
              <iodef:Address category="ipv4-addr">
                192.0.2.106
              </iodef:Address>
            </iodef:Node>
            <iodef:Service ip-protocol="6">
              <iodef:Port>1337</iodef:Port>
            </iodef:Service>
          </iodef:System>
          <iodef:System category="source" spoofed="yes">
            <iodef:Node>
              <iodef:Address category="ipv4-net">
                198.51.100.0/24
              </iodef:Address>
            </iodef:Node>
            <iodef:Service ip-protocol="6">
              <iodef:Port>1337</iodef:Port>
```

```
          </iodef:Service>
        </iodef:System>
        <iodef:System category="source" spoofed="yes">
          <iodef:Node>
            <iodef:Address category="ipv6-addr">
              2001:db8:dead:beef::1
            </iodef:Address>
          </iodef:Node>
          <iodef:Service ip-protocol="6">
            <iodef:Port>1337</iodef:Port>
          </iodef:Service>
        </iodef:System>
        <iodef:System category="target">
          <iodef:Node>
            <iodef:Address category="ipv4-addr">
              203.0.113.1
            </iodef:Address>
          </iodef:Node>
          <iodef:Service ip-protocol="6">
            <iodef:Port>80</iodef:Port>
          </iodef:Service>
        </iodef:System>
        <iodef:System category="sensor">
          <iodef:Node>
          </iodef:Node>
          <iodef:Description>
            Information provided in Flow class instance is from
            Inspection of traffic from network tap
          </iodef:Description>
        </iodef:System>
      </iodef:Flow>
      <iodef:Expectation action="other"/>
    </iodef:EventData>
    <iodef:IndicatorData>
      <iodef:Indicator>
        <iodef:IndicatorID name="csirt.example.com" version="1">
          G83345941
        </iodef:IndicatorID>
        <iodef:Description>
          User-Agent string
        </iodef:Description>
        <iodef:Observable>
          <iodef:BulkObservable type="http-user-agent">
            <iodef:BulkObservableList>
              user-agent="Mozilla/5.0 (Macintosh; U;
              Intel Mac OS X 10.5; en-US; rv:1.9.2.12)
              Gecko/20101026 Firefox/3.6.12">
            </iodef:BulkObservableList>
```

```
            </iodef:BulkObservable>
          </iodef:Observable>
        </iodef:Indicator>
      </iodef:IndicatorData>
    </iodef:Incident>
  </IODEF-Document>
```

B.3. Spear Phishing

   The spear-phishing test exchanged information that described a spear-
   phishing email, including DNS records and addresses about the sender,
   malicious attached file information, and email data. The IODEF
   version used for the data representation was based on [RFC7970].

```
 <?xml version="1.0" encoding="UTF-8"?>
 <IODEF-Document version="2.00"
                 xmlns="urn:ietf:params:xml:ns:iodef-2.0"
                 xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
   <iodef:Incident purpose="reporting">
     <iodef:IncidentID name="csirt.example.com">
       189601
     </iodef:IncidentID>
     <iodef:DetectTime>2013-01-04T08:06:12+00:00</iodef:DetectTime>
     <iodef:StartTime>2013-01-04T08:01:34+00:00</iodef:StartTime>
     <iodef:EndTime>2013-01-04T08:31:27+00:00</iodef:EndTime>
     <iodef:ReportTime>2013-01-04T09:15:45+00:00</iodef:ReportTime>
     <iodef:GenerationTime>2013-01-04T09:15:45+00:00
     </iodef:GenerationTime>
     <iodef:Description>
       Zeus Spear Phishing E-mail with Malware Attachment
     </iodef:Description>
     <iodef:Assessment occurrence="potential">
       <iodef:SystemImpact severity="medium" type="takeover-system">
         <iodef:Description>
           Malware with Command and Control Server and System Changes
         </iodef:Description>
       </iodef:SystemImpact>
     </iodef:Assessment>
     <iodef:Contact role="creator" type="organization">
       <iodef:ContactName>example.com CSIRT</iodef:ContactName>
       <iodef:Email>
         <iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
         </iodef:Email>
     </iodef:Contact>
     <iodef:EventData>
       <iodef:Description>
```

```
      Targeting Defense Contractors,
      specifically board members attending Dummy Con
    </iodef:Description>
    <iodef:Method>
      <iodef:Reference observable-id="ref-1234">
        <iodef:Description>Zeus</iodef:Description>
      </iodef:Reference>
    </iodef:Method>
    <iodef:Flow>
      <iodef:System category="source">
        <iodef:Node>
          <iodef:Address category="site-uri">
            http://www.zeusevil.example.com
          </iodef:Address>
          <iodef:Address category="ipv4-addr">
            192.0.2.166
          </iodef:Address>
          <iodef:Address category="asn">
            65535
          </iodef:Address>
          <iodef:Address category="ext-value"
                         ext-category="as-name">
            EXAMPLE-AS - University of Example
          </iodef:Address>
          <iodef:Address category="ext-value"
                         ext-category="as-prefix">
            192.0.2.0/24
          </iodef:Address>
        </iodef:Node>
        <iodef:NodeRole category="malware-distribution"/>
      </iodef:System>
    </iodef:Flow>
    <iodef:Flow>
      <iodef:System category="source">
        <iodef:Node>
          <iodef:DomainData>
            <Name>mail1.evildave.example.com</Name>
          </iodef:DomainData>
          <iodef:Address category="ipv4-addr">
            198.51.100.6
          </iodef:Address>
          <iodef:Address category="asn">
            65534
          </iodef:Address>
          <iodef:Address category="ext-value"
                         ext-category="as-name">
            EXAMPLE-AS - University of Example
          </iodef:Address>
```

```
            <iodef:DomainData>
              <iodef:Name>evildave.example.com</iodef:Name>
              <iodef:DateDomainWasChecked>2013-01-04T09:10:24+00:00
              </iodef:DateDomainWasChecked>
              <!-- <iodef:RelatedDNS RecordType="MX"> -->
              <iodef:RelatedDNS dtype="string">
                evildave.example.com MX preference = 10, mail exchanger
                = mail1.evildave.example.com
              </iodef:RelatedDNS>
              <iodef:RelatedDNS dtype="string">
                mail1.evildave.example.com
                internet address = 198.51.100.6
              </iodef:RelatedDNS>
              <iodef:RelatedDNS dtype="string">
                zuesevil.example.com. IN TXT \"v=spf1 a mx -all\"
              </iodef:RelatedDNS>
            </iodef:DomainData>
          </iodef:Node>
          <iodef:NodeRole category="mail">
            <iodef:Description>
              Sending phishing mails
            </iodef:Description>
          </iodef:NodeRole>
          <iodef:Service>
            <iodef:EmailData>
              <iodef:EmailFrom>
                emaildave@evildave.example.com
              </iodef:EmailFrom>
              <iodef:EmailSubject>
                Join us at Dummy Con
              </iodef:EmailSubject>
              <iodef:EmailX-Mailer>
                StormRider 4.0
              </iodef:EmailX-Mailer>
            </iodef:EmailData>
          </iodef:Service>
        </iodef:System>
        <iodef:System category="target">
          <iodef:Node>
            <iodef:Address category="ipv4-addr">
              203.0.113.2
            </iodef:Address>
          </iodef:Node>
        </iodef:System>
      </iodef:Flow>
      <iodef:Expectation action="other"/>
      <iodef:Record>
        <iodef:RecordData>
```

```
        <iodef:FileData observable-id="fd-1234">
          <iodef:File>
            <iodef:FileName>
              Dummy Con Sign Up Sheet.txt
            </iodef:FileName>
            <iodef:FileSize>
              152
            </iodef:FileSize>
            <iodef:HashData scope="file-contents">
              <iodef:Hash>
                <ds:DigestMethod Algorithm=
                "http://www.w3.org/2001/04/xmlenc#sha256"/>
                <ds:DigestValue>
                  141accec23e7e5157de60853cb1e01bc38042d
                  08f9086040815300b7fe75c184
                </ds:DigestValue>
              </iodef:Hash>
            </iodef:HashData>
          </iodef:File>
        </iodef:FileData>
      </iodef:RecordData>
      <iodef:RecordData>
        <iodef:CertificateData>
          <iodef:Certificate>
            <ds:X509Data>
              <ds:X509IssuerSerial>
                <ds:X509IssuerName>FakeCA
                </ds:X509IssuerName>
                <ds:X509SerialNumber>
                  57482937101
                </ds:X509SerialNumber>
              </ds:X509IssuerSerial>
              <ds:X509SubjectName>EvilDaveExample
              </ds:X509SubjectName>
            </ds:X509Data>
          </iodef:Certificate>
        </iodef:CertificateData>
      </iodef:RecordData>
    </iodef:Record>
  </iodef:EventData>
 </iodef:Incident>
</IODEF-Document>
```

B.4. Malware

   In this test, malware information was exchanged using RID and IODEF.
   The information included file hashes, registry setting changes, and
   the C2 servers the malware uses.

```
<?xml version="1.0" encoding="UTF-8"?>
<IODEF-Document version="2.00"
                xmlns="urn:ietf:params:xml:ns:iodef-2.0"
                xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <iodef:Incident purpose="reporting">
    <iodef:IncidentID name="csirt.example.com">
      189234
    </iodef:IncidentID>
    <iodef:ReportTime>2013-03-07T16:14:56.757+05:30</iodef:ReportTime>
    <iodef:GenerationTime>2013-03-07T16:14:56.757+05:30
    </iodef:GenerationTime>
    <iodef:Description>
      Malware and related indicators identified
    </iodef:Description>
    <iodef:Assessment occurrence="potential">
      <iodef:SystemImpact severity="medium" type="breach-proprietary">
        <iodef:Description>
          Malware with Command and Control Server and System Changes
        </iodef:Description>
      </iodef:SystemImpact>
    </iodef:Assessment>
    <iodef:Contact role="creator" type="organization">
      <iodef:ContactName>example.com CSIRT</iodef:ContactName>
      <iodef:Email>
        <iodef:EmailTo>contact@csirt.example.com</iodef:EmailTo>
      </iodef:Email>
    </iodef:Contact>
    <iodef:EventData>
      <iodef:Method>
        <iodef:Reference>
          <iodef:URL>
            http://www.threatexpert.example.com/report.aspx?
            md5=e2710ceb088dacdcb03678db250742b7
          </iodef:URL>
          <iodef:Description>Zeus</iodef:Description>
        </iodef:Reference>
      </iodef:Method>
      <iodef:Flow>
        <iodef:System category="source">
          <iodef:Node>
```

```
        <iodef:Address category="ipv4-addr"
                       observable-id="addr-c2-91011-001">
          203.0.113.200
        </iodef:Address>
        <iodef:Address category="site-uri"
                       observable-id="addr-c2-91011-002">
          http://zeus.556677889900.example.com/log-bin/
          lunch_install.php?aff_id=1&amp;
          lunch_id=1&amp;maddr=&amp;
          action=install
        </iodef:Address>
      </iodef:Node>
      <iodef:NodeRole category="c2-server"/>
    </iodef:System>
  </iodef:Flow>
  <iodef:Record>
    <iodef:RecordData>
      <iodef:FileData observable-id="file-91011-001">
        <iodef:File>
          <iodef:HashData scope="file-contents">
            <iodef:Hash>
              <ds:DigestMethod Algorithm=
                      "http://www.w3.org/2001/04/xmlenc#sha1"/>
              <ds:DigestValue>
                MHg2NzUxQTI1MzQ4M0E2N0Q4NkUwRjg0NzYwRjYxRjEwQkJKDQzJF
                REZG
              </ds:DigestValue>
            </iodef:Hash>
          </iodef:HashData>
        </iodef:File>
        <iodef:File>
          <iodef:HashData scope="file-contents">
            <iodef:Hash>
              <ds:DigestMethod Algorithm=
                      "http://www.w3.org/2001/04/xmlenc#md5"/>
              <ds:DigestValue>
                MHgyRTg4ODA5ODBENjI0NDdFOTc5MEFGQTg5NTEzRjBBNA==
              </ds:DigestValue>
            </iodef:Hash>
          </iodef:HashData>
        </iodef:File>
      </iodef:FileData>
      <iodef:WindowsRegistryKeysModified observable-id=
      "regkey-91011-001">
        <iodef:Key registryaction="add-value">
          <iodef:KeyName>
            HKLM\Software\Microsoft\Windows\
            CurrentVersion\Run\tamg
```

```
              </iodef:KeyName>
              <iodef:Value>
                ?\?\?%System%\wins\mc.exe\?\??
              </iodef:Value>
            </iodef:Key>
            <iodef:Key registryaction="modify-value">
              <iodef:KeyName>HKLM\Software\Microsoft\
                Windows\CurrentVersion\Run\dqo
              </iodef:KeyName>
              <iodef:Value>"\"\"%Windir%\Resources\
                Themes\Luna\km.exe\?\?"
              </iodef:Value>
            </iodef:Key>
          </iodef:WindowsRegistryKeysModified>
        </iodef:RecordData>
      </iodef:Record>
    </iodef:EventData>
    <iodef:EventData>
      <iodef:Method>
        <iodef:Reference>
          <iodef:URL>
            http://www.threatexpert.example.com/report.aspx?
            md5=c3c528c939f9b176c883ae0ce5df0001
          </iodef:URL>
          <iodef:Description>Cridex</iodef:Description>
        </iodef:Reference>
      </iodef:Method>
      <iodef:Flow>
        <iodef:System category="source">
          <iodef:Node>
            <iodef:Address category="ipv4-addr"
                          observable-id="addr-c2-91011-003">
              203.0.113.100
            </iodef:Address>
          </iodef:Node>
          <iodef:NodeRole category="c2-server"/>
          <iodef:Service ip-protocol="6">
            <iodef:Port>8080</iodef:Port>
          </iodef:Service>
        </iodef:System>
      </iodef:Flow>
      <iodef:Record>
        <iodef:RecordData>
          <iodef:FileData observable-id="file-91011-002">
            <iodef:File>
              <iodef:HashData scope="file-contents">
                <iodef:Hash>
```

```
                    <ds:DigestMethod Algorithm=
                            "http://www.w3.org/2001/04/xmlenc#sha1"/>
                    <ds:DigestValue>
                      MHg3MjYzRkUwRDNBMDk1RDU5QzhFMEM4OTVBBOUM
                      1ODVFMzQzRTcxNDFD
                    </ds:DigestValue>
                  </iodef:Hash>
                </iodef:HashData>
              </iodef:File>
            </iodef:FileData>
            <iodef:FileData observable-id="file-91011-003">
              <iodef:File>
                <iodef:HashData scope="file-contents">
                  <iodef:Hash>
                    <ds:DigestMethod Algorithm=
                            "http://www.w3.org/2001/04/xmlenc#md5"/>
                    <ds:DigestValue>
                      MHg0M0NEODUwRkNEQURFNDMzMEE1QkVBNkYxNkVFOTcxQw==
                    </ds:DigestValue>
                  </iodef:Hash>
                </iodef:HashData>
              </iodef:File>
            </iodef:FileData>
            <iodef:WindowsRegistryKeysModified observable-id=
                    "regkey-91011-002">
              <iodef:Key registryaction="add-value">
                <iodef:KeyName>
                  HKLM\Software\Microsoft\Windows\
                  CurrentVersion\Run\KB00121600.exe
                </iodef:KeyName>
                <iodef:Value>
                  \?\?%AppData%\KB00121600.exe\?\?
                </iodef:Value>
              </iodef:Key>
            </iodef:WindowsRegistryKeysModified>
          </iodef:RecordData>
        </iodef:Record>
      </iodef:EventData>
      <iodef:IndicatorData>
        <iodef:Indicator>
          <iodef:IndicatorID name="csirt.example.com" version="1">
            ind-91011
          </iodef:IndicatorID>
          <iodef:Description>
            evil c2 server, file hash, and registry key
          </iodef:Description>
          <iodef:IndicatorExpression operator="or">
            <iodef:IndicatorExpression operator="or">
```

```
        <iodef:Observable>
          <iodef:Address category="site-uri"
                         observable-id="addr-qrst">
            http://foo.example.com:12345/evil/cc.php
          </iodef:Address>
        </iodef:Observable>
        <iodef:Observable>
          <iodef:Address category="ipv4-addr"
                         observable-id="addr-stuv">
            192.0.2.1
          </iodef:Address>
        </iodef:Observable>
        <iodef:Observable>
          <iodef:Address category="ipv4-addr"
                         observable-id="addr-tuvw">
            198.51.100.1
          </iodef:Address>
        </iodef:Observable>
        <iodef:Observable>
          <iodef:Address category="ipv6-addr"
                         observable-id="addr-uvwx">
            2001:db8:dead:beef::1
          </iodef:Address>
        </iodef:Observable>
        <iodef:ObservableReference uid-ref="addr-c2-91011-001"/>
        <iodef:ObservableReference uid-ref="addr-c2-91011-002"/>
        <iodef:ObservableReference uid-ref="addr-c2-91011-003"/>
      </iodef:IndicatorExpression>
      <iodef:IndicatorExpression operator="and">
        <iodef:Observable>
          <iodef:FileData observable-id="file-91011-000">
            <iodef:File>
              <iodef:HashData scope="file-contents">
                <iodef:Hash>
                  <ds:DigestMethod Algorithm=
                        "http://www.w3.org/2001/04/xmlenc#sha256"/>
                  <ds:DigestValue>
                    141accec23e7e5157de60853cb1e01bc38042d08f
                    9086040815300b7fe75c184
                  </ds:DigestValue>
                </iodef:Hash>
              </iodef:HashData>
            </iodef:File>
          </iodef:FileData>
        </iodef:Observable>
        <iodef:Observable>
          <iodef:WindowsRegistryKeysModified observable-id=
                "regkey-91011-000">
```

```
                <iodef:Key registryaction="add-key"
                           observable-id="regkey-vwxy">
                  <iodef:KeyName>
                    HKLM\SYSTEM\CurrentControlSet\
                    Services\.Net CLR
                  </iodef:KeyName>
                </iodef:Key>
                <iodef:Key registryaction="add-key"
                           observable-id="regkey-wxyz">
                  <iodef:KeyName>
                    HKLM\SYSTEM\CurrentControlSet\
                    Services\.Net CLR\Parameters
                  </iodef:KeyName>
                  <iodef:Value>
                    \"\"%AppData%\KB00121600.exe\"\"
                  </iodef:Value>
                </iodef:Key>
                <iodef:Key registryaction="add-value"
                           observable-id="regkey-xyza">
                  <iodef:KeyName>
                    HKLM\SYSTEM\CurrentControlSet\Services\
                    .Net CLR\Parameters\ServiceDll
                  </iodef:KeyName>
                  <iodef:Value>C:\bad.exe</iodef:Value>
                </iodef:Key>
                <iodef:Key registryaction="modify-value"
                           observable-id="regkey-zabc">
                  <iodef:KeyName>
                    HKLM\SYSTEM\CurrentControlSet\
                    Services\.Net CLR\Parameters\Bar
                  </iodef:KeyName>
                  <iodef:Value>Baz</iodef:Value>
                </iodef:Key>
              </iodef:WindowsRegistryKeysModified>
            </iodef:Observable>
          </iodef:IndicatorExpression>
          <iodef:IndicatorExpression operator="or">
            <iodef:IndicatorExpression operator="and">
              <iodef:ObservableReference uid-ref="file-91011-001"/>
              <iodef:ObservableReference uid-ref="regkey-91011-001"/>
            </iodef:IndicatorExpression>
            <iodef:IndicatorExpression operator="and">
              <iodef:IndicatorExpression operator="or">
                <iodef:ObservableReference uid-ref="file-91011-002"/>
                <iodef:ObservableReference uid-ref="file-91011-003"/>
              </iodef:IndicatorExpression>
              <iodef:ObservableReference uid-ref="regkey-91011-002"/>
            </iodef:IndicatorExpression>
```

```
              </iodef:IndicatorExpression>
            </iodef:IndicatorExpression>
          </iodef:Indicator>
        </iodef:IndicatorData>
      </iodef:Incident>
</IODEF-Document>
```

B.5. IoT Malware

   The Internet of Things (IoT) malware test exchanged information that
   described a bad IP address of IoT malware and its scanned ports.
   This example information is extracted from alert messages of a
   darknet monitoring system referred to in [RFC8134]. The IODEF
   version used for the data representation was based on [RFC7970].

```
   <?xml version="1.0" encoding="UTF-8"?>
   <IODEF-Document version="2.00"
                   xmlns="urn:ietf:params:xml:ns:iodef-2.0"
                   xmlns:iodef="urn:ietf:params:xml:ns:iodef-2.0"
                   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
     <iodef:Incident purpose="reporting">
       <iodef:IncidentID name="csirt.example.com">
         189802
       </iodef:IncidentID>
       <iodef:ReportTime>2017-03-01T01:15:00+09:00</iodef:ReportTime>
       <iodef:GenerationTime>2017-03-01T01:15:00+09:00
       </iodef:GenerationTime>
       <iodef:Description>IoT Malware and related indicators
       </iodef:Description>
       <iodef:Assessment occurrence="potential">
         <iodef:SystemImpact severity="medium" type="takeover-system">
           <iodef:Description>IoT Malware is scanning other hosts
           </iodef:Description>
         </iodef:SystemImpact>
       </iodef:Assessment>
       <iodef:Contact role="creator" type="organization">
         <iodef:ContactName>example.com CSIRT
         </iodef:ContactName>
         <iodef:Email>
           <iodef:EmailTo>contact@csirt.example.com
           </iodef:EmailTo>
         </iodef:Email>
       </iodef:Contact>
       <iodef:EventData>
         <iodef:Discovery source="nidps">
           <iodef:Description>
             Detected by darknet monitoring
           </iodef:Description>
```

```
        </iodef:Discovery>
        <iodef:Flow>
          <iodef:System category="source">
            <iodef:Node>
              <iodef:Address category="ipv4-addr">
                192.0.2.210
              </iodef:Address>
            </iodef:Node>
            <iodef:NodeRole category="camera"/>
            <iodef:Service ip-protocol="6">
              <iodef:Port>23</iodef:Port>
            </iodef:Service>
            <iodef:OperatingSystem>
              <iodef:Description>
                Example Surveillance Camera OS 2.1.1
              </iodef:Description>
            </iodef:OperatingSystem>
          </iodef:System>
        </iodef:Flow>
        <iodef:EventData>
          <iodef:Flow>
            <iodef:System category="target">
              <iodef:Node>
                <iodef:Address category="ipv4-addr">
                  198.51.100.1
                </iodef:Address>
              </iodef:Node>
              <iodef:NodeRole category="honeypot"/>
              <iodef:Service ip-protocol="6">
                <iodef:Port>23</iodef:Port>
              </iodef:Service>
            </iodef:System>
          </iodef:Flow>
        </iodef:EventData>
        <iodef:EventData>
          <iodef:Flow>
            <iodef:System category="target">
              <iodef:Node>
                <iodef:Address category="ipv4-addr">
                  198.51.100.94
                </iodef:Address>
              </iodef:Node>
              <iodef:NodeRole category="honeypot"/>
              <iodef:Service ip-protocol="6">
                <iodef:Port>23</iodef:Port>
              </iodef:Service>
            </iodef:System>
          </iodef:Flow>
```

```
          </iodef:EventData>
          <iodef:EventData>
            <iodef:Flow>
              <iodef:System category="target">
                <iodef:Node>
                  <iodef:Address category="ipv4-addr">
                    198.51.100.237
                  </iodef:Address>
                </iodef:Node>
                <iodef:NodeRole category="honeypot"/>
                <iodef:Service ip-protocol="6">
                  <iodef:Port>2323</iodef:Port>
                </iodef:Service>
              </iodef:System>
            </iodef:Flow>
          </iodef:EventData>
        </iodef:EventData>
      </iodef:Incident>
    </IODEF-Document>
```

Authors' Addresses

   Panos Kampanakis
   Cisco Systems

   Email: pkampana@cisco.com

   Mio Suzuki
   NICT
   4-2-1, Nukui-Kitamachi
   Koganei, Tokyo 184-8795
   Japan

   Email: mio@nict.go.jp