MILE Working Group
Internet-Draft
Intended status: Experimental
Expires: August 16, 2014

D. Miyamoto
UTokyo
T. Takahashi
NICT

February 12, 2014

Knowledge obtained from the implementation experience of an IODEF-capable incident response management system
draft-daisuke-iodef-experiment-00.txt

Abstract

This document explains our observation on the usability of IODEF [RFC5070], based on our experiments. We aim at developing an IODEF-capable incident response management systems in order to facilitate incident response activities. We started to design and implement the system for our university CERT, however, there are several technical issues while implementing and operating the system. This document shares the observation from our proto-type implementation and provides new sight from operational aspects.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The number of incidents in cyber society is growing day by day. Incident information needs to be reported, exchanged, and shared among organizations in order to cope with the situation. IODEF provides a scheme to describe and exchange incident response information among interested parties.

For our university CERT, we decided to introduce an IODEF-capable incident response management system to facilitate incident response activities. Our university has two types of CERT, namely, a central CERT and divisional CERTs. The former is a contact point for external organizations, and the latter is a CERT for each division in the university. When the central CERT receives such information, it notifies the information to the corresponding divisional CERT who has an accountability for decision and actions.

Our old system employed emails for exchanging the information between the central and divisional CERTS, however, we started to employ machine-readable message in regard to the growing demand for automated incident response systems. For doing so, we attempted to implement an IODEF-capable incident response management system.

In our implementation, we encountered problems while dealing with XML schema. To save the development cost, we employed code generators that build class libraries for accessing values in IODEF elements. Due to the complexity of IODEF message format defined in [RFC5070], some code generators could not understand its schema.

We also found some operational problems as well as the implementation problem. Most of the problems were on the choice of values for IODEF attributes and/or elemens.

This draft provides how we evade the implementation problem, and explores the suitable value for XML element in regard to the incidents.

2. Terminology

The terminology used in this document follows the one defined in [RFC5070].

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].

3. Implementation

Since a code generator for XSD automatically develops useful libraries for accessing XML attributes and/or composing messages, we tested following generators to build the libraries from RFC 5070 [RFC5070] .

- XML::Pastor [XSD:Perl] (Perl)
- RXSD [XSD:Ruby] (Ruby)
- PyXB [XSD:Python] (Python)
- JAXB [XSD:Java] (Java)
- CodeSynthesis XSD [XSD:Cxx] (C++)
- Xsd.exe [XSD:CS] (C#)

We thought we can use them to generate IODEF, but they cannot be easily used. For instance, we have used XML::Pastor, but it could not properly understand its schema due to the complexity of IODEF XSD. The same applies to RXSD and JAXB. Only PyXB, CodeSynthesis XSD and Xsd.exe were able to understand the schema.

To cope with the situation, we have made a trick, which is not recommended, but is one option to go through the situation. That is, "XSD2XML2XSD", which means that XSD is converted to XML, and it is again converted to XSD. The resultant XSD was process-able by the all tools above.

Nevertheless, the generated module was unworkable. This is due to the fact that IODEF uses '-' (hyphen) symbols in its classes or attributes, listed as follows.

- o IODEF-Document Class; it is the top level class in the IODEF data model described in section 3.1 of [RFC5070].
- o The vlan-name and vlan-num Attribute; according to section 3.16.2 of [RFC5070], they are the name and number of Virtual LAN and are the attributes for Address class.
- o Extending the Enumerated Values of Attribute; according to section 5.1 of [RFC5070], it is a extension techniques to add new enumerated values to an attribute, and has a prefix of "ext-", e.g., ext-value, ext-category, ext-type, and so on.

According to the language specification, Perl classes and/or functions could not contain '-' symbols in their names. We replaced hyphens with '_' (underscore) symbols to evade this issue. Before outputting an IODEF format message, our system must manually replace these renamed characters in its serialization process.

Aside from the case of Perl, other language tend to evade using any hyphens in its name space. PyXB and CodeSynthesis XSD automatically replaced hyphen with underscore symbols, and JAXB and Xsd.exe simply removed hyphens. These tools also might output an exact IODEF message format through their serialization process. RXSD was similar to JAXB and Xsd.exe, replaced with hyphens automatically, but did not support converting the renamed characters for outputting.

4. Operational Issues

This section explains some pitfalls while assigning values for IODEF-based XML elements. Mainly, our central CERT notifies the incident information to the issued divisional CERT, and the divisional CERT reports the results of forensics. Based on this situation, we found several cases that we were not sure about which attributes should be chosen.

4.1. type attribute @ Impact class

Various incident classification exist. For instance, JPCERT proposes the following classification: phishing site, page hijack, malware propagation, scan, DoS/DDoS, and control systems. Nevertheless, it is hard to fit them into the type attribute of theimpact class.

For example, phishing site, scan, and DoS/DDoS might be mapped as "social-engineering", "recon", and "dos" attributes in respectively. In the rest of cases, what the type of the attribute should we choose?

4.2. category attribute @ NodeRole Class

IODEF has category attribute for NodeRole class. Though various categories are described, they are not enough. For instance, we sometime report the category of "proxy server" in our daily CERT operation, but which one am we supposed to choose? How about web mail? Should we choose "www"? or "mail"?

4.3. action attribute @ Expectation Class

Assuming if the notifier sends a message with expecting to forensic for the issues, and the reporter answers the result of their forensics. In such cases, the notifiers would choose "investigation", but what types of action attribute should the reporter choose? Should the reporter choose "nothing" ?

When a notifier sends IODEF document, the report wishes to confirm it without asking any further actions. Then what values shall we choose?

4.4. Potential information leakage

The numbering of Incident ID needs to be considered. Otherwise, information, such as the number of incidents within certain period could be observed by document receivers. For instance, we could randomize the assignment of the numbers.

4.5. Configuration of Nodes

Node class can describe various information of the system, but the level of information granularity there is not defined. It could be that very detailed information is needed, or it could be the opposite. It has the field of the software id and configid, but the formats for them are not specifically defined.

It is natural to guess that we cannot define single, common level of information granularity. Depending on situation and operation, the needed level of information granularity differs.

Thus one approach is using IODEF-SCI, which can choose arbitrary schema to describe the details of such information.

5. Security Considerations

This document raises no security issues itself. The potential security issues are the vulnerabilities in the class libraries constructed by code generators.

6. IANA Considerations

This document contains no considerations for IANA.

7. Conclusions

The document explains the implemetation issue, the problems raised from code generation, and the operational issue, the problems while choosing the value in XML elements for IODEF format messages.

8. Acknowledgements

Many thanks for feedback from Tomohiro Ishihara for his comments. This work is materially supported by the Ministry of Internal Affairs and Communication, Japan, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No. 608533 (NECOMA).

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, December 2007.

9.2. Informative References

[XSD:Perl]
Ulsoy, A., "XML::Pastor", <http://search.cpan.org/~aulusoy/XML-Pastor-1.0.4/>.

[XSD:Ruby]
Morsi, M., "RXSD - XSD / Ruby Translator", <https://github.com/movitto/RXSD>.

[XSD:Python]
Bigot, P., "PyXB: Python XML Schema Bindings", <https://pypi.python.org/pypi/PyXB>.

[XSD:Java]
Project Kenai, "JAXB Reference Implementation", <https://jaxb.java.net/>.

[XSD:Cxx] CodeSynthesis, "XSD - XML Data Binding for C++", <http://www.codesynthesis.com/>.

[XSD:CS] Microsoft, "XML Schema Definition Tool (Xsd.exe)", <http://www.codesynthesis.com/>.

Authors' Addresses

Daisuke Miyamoto
The University of Tokyo
2-11-16 Yayoi Bunkyo-Ku
113-8658 Tokyo
Japan

Phone: +80 3 5841 0836
Email: daisu-mi@nc.u-tokyo.ac.jp

Takeshi Takahashi
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi Koganei
184-8795 Tokyo
Japan

Phone: +80 423 27 5862
Email: takeshi_takahashi@nict.go.jp