Internet Engineering Task Force (IETF)
Request for Comments: 8134
Category: Informational
ISSN: 2070-1721

C. Inacio
CMU
D. Miyamoto
UTokyo

May 2017

Management Incident Lightweight Exchange (MILE) Implementation Report

Abstract

This document is a collection of implementation reports from vendors, consortiums, and researchers who have implemented one or more of the standards published from the IETF INCident Handling (INCH) and Management Incident Lightweight Exchange (MILE) working groups.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc8134.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
2. Consortiums and Information Sharing and Analysis Centers (ISACs) 4
2.1. Anti-Phishing Working Group 4
2.2. Advanced Cyber Defence Centre 4
2.3. Research and Education Networking Information Sharing and Analysis Center 4
3. Open Source Implementations 4
3.1. EMC/RSA RID Agent 4
3.2. NICT IODEF-SCI implementation 5
3.3. n6 5
4. Vendor Implementations 6
4.1. Deep Secure 6
4.2. IncMan Suite, DFLabs 7
4.3. Surevine Proof of Concept 8
4.4. MANTIS Cyber-Intelligence Management Framework 8
5. Vendors with Planned Support 9
5.1. Threat Central, HP 9
5.2. DAEDALUS, NICT 9
6. Other Implementations 10
6.1. Collaborative Incident Management System 10
6.2. Automated Incident Reporting - AirCERT 10
6.3. US Department of Energy CyberFed 11
7. Implementation Guide 11
7.1. Code Generators 11
7.2. iodeflib 13
7.3. iodefpm 13
7.4. Usability 13
8. IANA Considerations 14
9. Security Considerations 14
10. Informative References 14
Acknowledgements 16
Authors' Addresses 16

1. Introduction

This document is a collection of information about security incident reporting protocols and the implementation of systems that use them to share such information. It is simply a collection of information, and it makes no attempt to compare the various standards or implementations. As such, it will be of interest to network operators who wish to collect and share such data.

Operationally, operators would need to decide which incident data collection group they want to be part of, and that choice will strongly influence their choice of reporting protocol and applications used to gather and distribute the data.

This document is a collection of implementation reports from vendors and researchers who have implemented one or more of the standards published from the INCH and MILE working groups. The standards include:

- Incident Object Description Exchange Format (IODEF) v1 [RFC5070]

- Incident Object Description Exchange Format (IODEF) v2 [RFC7970]

- Extensions to the IODEF-Document Class for Reporting Phishing [RFC5901]

- Sharing Transaction Fraud Data [RFC5941]

- Real-time Inter-network Defense (RID) [RFC6545]

- Transport of Real-time Inter-network Defense (RID) Messages over HTTP/TLS [RFC6546]

- Incident Object Description Exchange Format (IODEF) Extension for Structured Cybersecurity Information (SCI) [RFC7203]

The implementation reports included in this document have been provided by the team or product responsible for the implementations of the mentioned RFCs. A more complete list of implementations, including open source efforts and vendor products, can also be found at the following location:

<http://siis.realmv6.org/implementations/>

2. Consortiums and Information Sharing and Analysis Centers (ISACs)

2.1. Anti-Phishing Working Group

The Anti-Phishing Working Group (APWG) is one of the biggest coalitions against cybercrime, especially phishing. In order to collect threat information in a structured format, APWG provides a phishing and cybercrime reporting tool that sends threat information to APWG by tailoring information with the IODEF format, based on RFC 5070 [RFC5070] and RFC 5901 [RFC5901].

2.2. Advanced Cyber Defence Centre

The Advanced Cyber Defence Centre (ACDC) is a Europe-wide activity to fight against botnets. ACDC provides solutions to mitigate on-going attacks and consolidates information provided by various stakeholders into a pool of knowledge. Within ACDC, IODEF is one of the supported schemas for exchanging the information.

2.3. Research and Education Networking Information Sharing and Analysis Center

The Research and Education Networking Information Sharing and Analysis Center (REN-ISAC) is a private community of researchers and higher-education members that share threat information and employs IODEF formatted-messages to exchange information.

REN-ISAC also recommends using an IODEF attachment provided with a notification email for processing rather than relying on parsing of the body text of email. The tools provided by REN-ISAC are designed to handle such email.

<http://www.ren-isac.net/notifications/using_iodef.html>

3. Open Source Implementations

3.1. EMC/RSA RID Agent

The EMC/RSA RID agent is an open source implementation of the IETF standards for the exchange of incident and indicator data. The code has been released under an MIT license, and development will continue with the open source community at the GitHub site for RSA Intelligence Sharing:

<https://github.com/RSAIntelShare/RID-Server.git>

The code implements the Real-time Inter-network Defense (RID) described in RFC 6545 [RFC6545] and the Transport of RID over HTTP/TLS protocol described in [RFC6546]. The code supports the evolving Incident Object Description Exchange Format (IODEF) data model [RFC7970] from the work in the IETF Managed Incident Lightweight Exchange (MILE) working group.

3.2. NICT IODEF-SCI implementation

Japan's National Institute of Information and Communications Technology (NICT) Network Security Research Institute implemented open source tools for exchanging, accumulating, and locating IODEF-SCI [RFC7203] documents.

Three tools are available from GitHub. These tools assist the exchange of IODEF-SCI documents between parties. IODEF-SCI [RFC7203] extends IODEF so that an IODEF document can embed Structured Cybersecurity Information (SCI). For instance, it can embed Malware Metadata Exchange Format (MMDEF), Common Event Expression (CEE), Malware Attribute Enumeration and Characterization (MAEC) in XML, and Common Vulnerabilities and Exposures (CVE) identifiers.

The three tools are generator, exchanger, and parser. The generator generates IODEF-SCI documents or appends XML to an existing IODEF document. The exchanger sends the IODEF document to a specified correspondent node. The parser receives, parses, and stores the IODEF-SCI document. The parser also creates an interface that enables users to locate IODEF-SCI documents that have previously been received. The code has been released under an MIT license and development will continue on GitHub.

Note that users can enjoy using this software at their own risk.

Available Online:

<https://github.com/TakeshiTakahashi/IODEF-SCI>

3.3. n6

n6 is a platform for processing security-related information; it was developed by the Poland Research and Academic Computer Network (NASK) Computer Emergency Response Team (CERT) Polska. The n6 API provides a common and unified way of representing data across the different sources that participate in knowledge management.

n6 exposes a REST-ful (Representational State Transfer) API over HTTPS with mandatory authentication via Transport Layer Security (TLS) client certificates to ensure confidential and trustworthy

communications. Moreover, it uses an event-based data model for representation of all types of security information.

Each event is represented as a JSON object with a set of mandatory and optional attributes. n6 also supports alternative output data formats for keeping compatibility with existing systems - IODEF and CSV - although these formats lack some of the attributes that may be present in the native JSON format.

Available Online:

<https://github.com/CERT-Polska/n6sdk>

4. Vendor Implementations

4.1. Deep Secure

Deep-Secure Guards are built to protect a trusted domain from:

- o releasing sensitive data that does not meet the organizational security policy, and

- o applications receiving badly constructed or malicious data that could exploit a vulnerability (known or unknown).

Deep-Secure Guards support HTTPS and the Extensible Messaging and Presence Protocol (XMPP -- optimized server-to-server protocol), transports. The Deep-Secure Guards support transfer of XML-based business content by creating a schema to translate the known good content to and from the intermediate format. This means that the Deep-Secure Guards can be used to protect:

- o IODEF/RID using the HTTPS transport binding [RFC6546]

- o IODEF/RID using an XMPP binding

- o Resource-Oriented Lightweight Indicator Exchange (ROLIE) using HTTPS transport binding [XEP-0268]

- o Structured Threat Information Expression (STIX) / Trusted Automated Exchange of Indicator Information (TAXII) using the HTTPS transport binding

Deep-Secure Guards also support the SMTP transport and perform deep content inspection of content including XML attachments. The Mail Guard supports S/MIME, and Deep Secure is working on support for the upcoming PLASMA standard, which enables an information-centric policy enforcement of data use.

4.2. IncMan Suite, DFLabs

The Incident Object Description Exchange Format, documented in RFC 5070 [RFC5070], defines a data representation that provides a framework for sharing information commonly exchanged by Computer Security Incident Response Teams (CSIRTs) about computer security incidents. IncMan Suite implements the IODEF standard for exchanging details about incidents, either for exporting or importing activities. This has been introduced to enhance the capabilities of the various CSIRTs to facilitate collaboration and sharing of useful experiences (sharing awareness on specific cases).

The IODEF implementation is specified as an XML schema; therefore all data are stored in an XML file. In this file, all the data of an incident are organized in a hierarchical structure to describe the various objects and their relationships.

The IncMan Suite relies on IODEF as a transport format, which is composed by various classes for describing the entities that are part of the incident description. For instance, the various relevant timestamps (detection time, start time, end time, and report time), the techniques used by the intruders to perpetrate the incident, the impact of the incident, technical and non-technical (time and monetary), and obviously all systems involved in the incident.

4.2.1. Exporting Incidents

Each incident defined in the IncMan Suite can be exported via a user interface feature, and it will create an XML document. Due to the nature of the data processed, the IODEF extraction might be considered privacy sensitive by the parties exchanging the information or by those described by it. For this reason, specific care needs to be taken in ensuring the distribution to an appropriate audience or third party, either during the document exchange or the subsequent processing.

The XML document generated will include a description and details of the incident along with all the systems involved and the related information. At this stage, it can be distributed for import into a remote system.

4.2.2. Importing Incidents

The IncMan Suite provides the functionality to import incidents stored in files and transported via IODEF-compliant XML documents. The importing process is comprised of two steps: first, the file is inspected to validate if it is well formed; second, all data are uploaded inside the system.

If the incident already exists in the system with the same incident ID, the new one being imported will be created under a new ID. This approach prevents accidentally overwriting existing information or merging inconsistent data.

The IncMan Suite also includes a feature to upload incidents from emails.

The incident, described in XML format, can be stored directly into the body of the email message or transported as an attachment of the email. At regular intervals that are customizable by the user, the IncMan Suite monitors for incoming emails, which are filtered by a configurable white-list and black-list mechanism on the sender's email account. Then, a parser processes the received email and a new incident is created automatically after having validated the email body or the attachment to ensure the format is well formed.

4.3. Surevine Proof of Concept

XMPP is enhanced and extended through the XMPP Extension Protocols (XEPs). XEP-0268 [XEP-0268] describes incident management (using IODEF) of the XMPP network itself, effectively supporting self-healing the XMPP network. In order to more generically cover the incident management of a network over the same network, XEP-0268 requires some updates. We are working on these changes together with a new XEP that supports "social networking" over XMPP, which enhances the publish-and-subscribe XEP [XEP-0060]. This now allows nodes to publish and subscribe to any type of content and therefore receive the content. XEP-0060 will be used to describe IODEF content. We now have an alpha version of the server-side software and client-side software required to demonstrate the "social networking" capability and are currently enhancing this to support cyber incident management in real time.

4.4. MANTIS Cyber-Intelligence Management Framework

Model-based Analysis of Threat Intelligence Sources (MANTIS) provides an example implementation of a framework for managing cyber threat intelligence expressed in standards such as STIX, Cyber Observable Expression (CybOX), IODEF, etc. The aims of providing such an example implementation are as follows:

- To facilitate discussions about emerging standards such as STIX, CybOX, et al., with respect to questions regarding tooling: how would a certain aspect be implemented, and how do changes affect an implementation? Such discussions become much easier and have a better basis if they can be lead in the context of example tooling that is known to the community.

- To lower the barrier of entry for organizations and teams (especially CSIRT/CERT teams) in using emerging standards for cyber-threat-intelligence management and exchange.

- To provide a platform on the basis of which research and community-driven development in the area of cyber-threat-intelligence management can occur.

5. Vendors with Planned Support

5.1. Threat Central, HP

HP has developed HP Threat Central, a security intelligence platform that enables automated, real-time collaboration between organizations to combat today's increasingly sophisticated cyber attacks. One way automated sharing of threat indicators is achieved is through close integration with the HP ArcSight Security Information and Event Management (SIEM) for automated upload and consumption of information from the Threat Central Server. In addition, HP Threat Central supports open standards for sharing threat information so that participants who do not use HP Security Products can participate in the sharing ecosystem. It is planned that future versions will also support IODEF for the automated upload and download of threat information.

5.2. DAEDALUS, NICT

DAEDALUS is a real-time alert system based on a large-scale darknet monitoring facility that has been deployed as a part of the Network Incident analysis Center for Tactical Emergency Response (nicter) system of NICT, which is based in Japan. DAEDALUS consists of an analysis center (i.e., nicter) and several cooperative organizations. Each organization installs a darknet sensor and establishes a secure channel between it and the analysis center, and it continuously forwards darknet traffic toward the center. In addition, each organization registers the IP address range of its livenet at the center in advance. When these distributed darknet sensors observe malware activities from the IP address of a cooperating organization, then the analysis center sends an alert to the organization. The future version of DAEDALUS will support IODEF for sending alert messages to the users.

6. Other Implementations

6.1. Collaborative Incident Management System

A Collaborative Incident Management System (CIMS) is a proof-of-concept system for collaborative incident handling and for the sharing of information about cyber defense situational awareness between the participants; it was developed for the Cyber Coalition 2013 (CC13) exercise organized by the North Atlantic Treaty Organization (NATO). CIMS was implemented based on Request Tracker (RT), an open source software widely used for handling incident responses by many CERTs and CSIRTs.

One of the functionalities implemented in CIMS was the ability to import and export IODEF messages in the body of emails. The intent was to verify the suitability of IODEF to achieve the objective of collaborative incident handling. The customized version of RT could be configured to send an email message containing an IODEF message whenever an incident ticket was created, modified, or deleted. These IODEF messages would then be imported into other incident handling systems in order to allow participating CSIRTs to use their usual means for incident handling while still interacting with those using the proof-of-concept CIMS. Having an IODEF message generated for every change made to the incident information in RT (and for the system to allow incoming IODEF email messages to be associated to an existing incident) would in some way allow all participating CSIRTs to actually work on a "common incident ticket", at least at the conceptual level. Of particular importance was the ability for users to exchange information between each other concerning actions taken in the handling of a particular incident, thus creating a sort of common action log as well as requesting/tasking others to provide information or perform a specified action and correlating received responses to the original request or task. As well, a specific "profile" was developed to identify a subset of the IODEF classes that would be used during the exercise in an attempt to channel all users into a common usage pattern of the otherwise flexible IODEF standard.

6.2. Automated Incident Reporting - AirCERT

AirCERT was implemented by the CERT / Coordination Center (CC) of Carnegie Mellon’s Software Engineering Institute CERT division. AirCERT was designed to be an Internet-scalable distributed system for sharing security event data. The AirCERT system was designed to be an automated collector of flow and Intrusion Detection System (IDS) alerts. AirCERT would collect that information into a relational database and be able to share reporting using IODEF and the Intrusion Detection Message Exchange Format [RFC4765]. AirCERT

additionally used SNML [SNML] to exchange information about the network. AirCERT was implemented in a combination of C and Perl modules and included periodic graphing capabilities leveraging the Round-Robin Database Tool (RRDTool).

AirCERT was intended for large-scale distributed deployment and, eventually, the ability to sanitize data to be shared across administrative domains. The architecture was designed to allow collection of data on a per-site basis and to allow each site to create data sharing based on its own particular trust relationships.

6.3. US Department of Energy CyberFed

The CyberFed system was implemented and deployed by Argonne National Laboratory to automate the detection and response of attack activity against Department of Energy (DoE) computer networks. CyberFed automates the collection of network alerting activity from various perimeter network defenses and logs those events into its database. CyberFed then automatically converts that information into blocking information transmitted to all participants. The original implementation used IODEF messages wrapped in an XML extension to manage a large array of indicators. The CyberFed system was not designed to describe a particular incident as much as to describe a set of current network-blocking indicators that can be generated and deployed machine to machine.

CyberFed is primarily implemented in Perl. Included as part of the CyberFed system are scripts that interact with a large number of firewalls, IDS / Intrusion Prevention System (IPS) devices, DNS systems, and proxies that operate to implement both the automated collection of events as well as the automated deployment of black listing.

Currently, CyberFed supports multiple exchange formats including IODEF and STIX. Open Indicators of Compromise (OpenIOC) is also a potential exchange format that the US DoE is considering.

7. Implementation Guide

The section aims at sharing tips for development of IODEF-capable systems.

7.1. Code Generators

For implementing IODEF-capable systems, it is feasible to employ code generators for the XML Schema Definition (XSD). The generators are used to save development costs since they automatically create useful libraries for accessing XML attributes, composing messages, and/or

validating XML objects. The IODEF XSD was defined in Section 8 of RFC 5070 [RFC5070] and is available from the "ns" registry <https://www.iana.org/assignments/xml-registry>.

However, some issues remain. Due to the complexity of the IODEF XSD, some code generators could not generate code from the XSD file. The tested code generators are as follows.

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

For instance, we have tried to use XML::Pastor, but it could not properly understand its schema due to the complexity of IODEF XSD. The same applies to Ruby XSD (RXSD) and Java Architecture for XML Binding (JAXB). Only Python XML Schema Bindings (PyXB), CodeSynthesis XSD, and Xsd.exe were able to understand the complex schema.

Unfortunately, there is no recommended workaround. A possible workaround is a double conversion of the XSD file. This entails the XSD being serialized into XML; afterwards, the resulting XML is converted back into an XSD. The resultant XSD was successfully processed by all the tools listed above.

It should be noted that IODEF uses '-' (hyphen) symbols in its classes or attributes, which are listed as follows:

- o IODEF-Document Class: It is the top-level class in the IODEF data model described in Section 3.1 of RFC 5070 [RFC5070].
- o The vlan-name and vlan-num Attributes: According to Section 3.16.2 of RFC 5070 [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 RFC 5070 [RFC5070], this is an extension technique to add new enumerated values to an attribute, and it has a prefix of "ext-", e.g., ext-value, ext-category, ext-type, and so on.

According to the language specification, many programming languages prohibit having '-' symbols in the name of class. The code generators must replace or remove the '-' when building the libraries. They should have the name space restore the '-' when outputting the XML along with IODEF XSD.

7.2. iodeflib

iodeflib is an open source implementation written in Python. This provides simple but powerful APIs to create, parse, and edit IODEF documents. It was designed in order to keep its interface as simple as possible, whereas generated libraries tend to inherit the complexity of IODEF XSD. In addition, the iodeflib interface includes functions to hide some unnecessarily nested structures of the IODEF schema and add more convenient shortcuts.

This tool is available through the following link:

<http://www.decalage.info/python/iodeflib>

7.3. iodefpm

IODEF.pm is an open source implementation written in Perl. This also provides a simple interface for creating and parsing IODEF documents in order to facilitate the translation of the key-value-based format to the IODEF representation. The module contains a generic XML DTD parser and includes a simplified node-based representation of the IODEF DTD. Hence, it can easily be upgraded or extended to support new XML nodes or other DTDs.

This tool is available through the following link:

<http://search.cpan.org/~saxjazman/>

7.4. Usability

Some tips to avoid problems are noted here:

- IODEF has a category attribute for the NodeRole class. Though various categories are described, they are not sufficient. For example, in the case of webmail servers, should the user choose "www" or "mail"? One suggestion is to select "mail" as the category attribute and add "www" for another attribute.

- The numbering of incident IDs needs to be considered. Otherwise, information, such as the number of incidents within a certain period, could be observed by document receivers. This is easily mitigated by randomizing the assignment of incident IDs.

8. IANA Considerations

This memo does not require any IANA actions.

9. Security Considerations

This document provides a summary of implementation reports from researchers and vendors who have implemented RFCs and drafts from the MILE and INCH working groups. There are no security considerations added because of the nature of the document.

10. Informative References

[RFC4765] Debar, H., Curry, D., and B. Feinstein, "The Intrusion Detection Message Exchange Format (IDMEF)", RFC 4765, DOI 10.17487/RFC4765, March 2007, <http://www.rfc-editor.org/info/rfc4765>.

[RFC5070] Danyliw, R., Meijer, J., and Y. Demchenko, "The Incident Object Description Exchange Format", RFC 5070, DOI 10.17487/RFC5070, December 2007, <http://www.rfc-editor.org/info/rfc5070>.

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

[RFC5941] M'Raihi, D., Boeyen, S., Grandcolas, M., and S. Bajaj, "Sharing Transaction Fraud Data", RFC 5941, DOI 10.17487/RFC5941, August 2010, <http://www.rfc-editor.org/info/rfc5941>.

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

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

[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, <http://www.rfc-editor.org/info/rfc7203>.

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

[SNML] Trammell, B., Danyliw, R., Levy, S., and A. Kompanek, "AirCERT: The Definitive Guide", 2005, <http://aircert.sourceforge.net/docs/aircert_manual-06_2005.pdf>.

[XEP-0060] Millard, P., Saint-Andre, P., and R. Meijer, "XEP-0060: Publish-Subscribe", December 2016, <http://www.xmpp.org/extensions/xep-0060.html>.

[XEP-0268] Hefczy, A., Jensen, F., Remond, M., Saint-Andre, P., and M. Wild, "XEP-0268: Incident Handling", May 2012, <http://xmpp.org/extensions/xep-0268.html>.

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

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

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

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

[XSD:Python] Bigot, P., "PyXB 1.2.5: Python XML Schema Bindings", <https://pypi.python.org/pypi/PyXB>.

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

Acknowledgements

The MILE implementation report has been compiled through the submissions of implementers of INCH and MILE working group standards. A special note of thanks to the following contributors:

John Atherton, Surevine

Humphrey Browning, Deep-Secure

Dario Forte, DFLabs

Tomas Sander, HP

Ulrich Seldeslachts, ACDC

Takeshi Takahashi, National Institute of Information and Communications Technology Network Security Research Institute

Kathleen Moriarty, EMC

Bernd Grobauer, Siemens

Dandurand Luc, NATO

Pawel Pawlinski, NASK

Authors' Addresses

Chris Inacio
Carnegie Mellon University
4500 5th Ave., SEI 4108
Pittsburgh, PA 15213
United States of America

Email: inacio@andrew.cmu.edu

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

Email: daisu-mi@nc.u-tokyo.ac.jp