eCSIRT.net Deliverable[1] Guideline to Application of the Common Language part (ii) — an Outline of the eCSIRT.net Toolkit

Deliverable D2.3.2/D3.2.1/D3.2.3 part (ii)[2]

Arne Helme
Stelvio
the Netherlands

Version 3.1, December 2003

[1]This work is funded by the European Commision's Fifth Framework Program, within the IST (Information Society Technology) branch, contract no. IST-2001-37558.

[2]Part (i) of this deliverable is contained in the Common Language Specification

Abstract

This document is part (ii) of a WP2/3 deliverable, the guideline on how to apply the common language. Part (i) is contained in the common langauge specification itself. This part (ii) describes the way that the toolkit developed within eCSIRT.net works. This toolkit had to be developed because of lack of readily available tools. The toolkit comprises two tools essentially, IHSH — a command line based IODEF object create/change/read tool, that can be integrated with existing incident databases, or used standalone — and a Perl library — especially meant to integrate with RT incident databases, but much wider useabel because of the flexibility it offers. All partners within the eCSIRT.net project used either IHSH, or the Perl library.

Contents

1 Overview

This document describes the way that the toolkit developed within eCSIRT.net works. The toolkit had to be developed because of lack of readily available tools. The toolkit comprises two tools essentially, IHSH — a command line based IODEF object create/change/read tool, that can be integrated with existing incident databases, or used stand-alone — and a Perl library — especially meant to integrate with RT incident databases, but much wider useabel because of the flexibility it offers. All partners within the eCSIRT.net project used either IHSH, or the Perl library.

2 Operational Framework

In order to contain and prevent security incidents, CSIRTs need to have a common understanding of such incidents and agreed-on operational procedures of exchanging and handling information.

Operational frameworks for CSIRTs are well documented in literature (see, e.g., the CSIRT handbook [West-Brown et al., 1998]). The definition of a computer security incident response team used in this paper is the one devised by [Koek et al., 2001]:

> If an entity A entertains a function B where customers/constituencies can report computer/network security incidents, and B then handles these reports in a constructive and secure way (consulting, coordination, feedback, . . .), then function B essentially is the CSIRT of entity A.

For eCSIRT.net purposes a certain similarity in purpose and operation of the participating CSIRTs is necessary, for the exchange of incident data to be successful and meaningful. This necessary similarity is ensured by only allowing teams in that are TI accredited. TI accreditation is seen as sufficient condition - a necessary condition has not been defined yet.

Additionally eCSIRT.net partners will have to sign a “Code of Conduct” before embarking on standardized data exchange — which together with TI accreditation further defines the eCSIRT.net operational framework.

3 Integration Tools

3.1 IHSH

3.1.1 Overview

IHSH is a small LibAir-based command-line tool manage (create, read, write, and experiment with) XML formatted IODEF messages. It is based on the LibAir library developed at CERT/CC, and implements a small command interpreter that can be invoked from Unix SHell scripts. Using IHSH's command IODEF messages can easily be created or modified. Using an XPAT-like syntax, it is possible to "walk" the internal data representation of IODEF messages in order to set/get values and attributes. IHSH also contains basic support to communicate natively with an ARS/Remedy server. At the time of writing the ARS code is experimental.

IHSH can be used in two different modes of operations either to parse incident reports encoded in XML according to the IODEF specification, or to generate an XML formatted report using IODEF comliant directives provided by an external application. In the former case, the output of IHSH is a list of commands that an application can use to drive a user interface to illustrate the report. In the latter case, the output is an XML formatted report.

In addition, IHSH can be used as a shell to create incident reports manually. The functionality can be used to manually edit an incident report and change some of its contents.

When compiled with ARS/Remedy support, IHSH communicates directly with an ARS server and can fetch existing data from the server or update the server with new information.

3.1.2 Functions

IHSH understand a simple command interpreter syntax that closely resembles the data handling primitives provided by the underlying LibIH library. Given a tree representation of a document, a weak XPat-like syntax is used to locate a given node (XML element) of interest in the tree. The following commands are recognized:

`cd` <*path*>**:** Conduct a tree walk to the location specified in *path*.

`fsetval` <*fname*>**:** At the current location in the tree, set attribute *name* to the contents of the file *fname*. This command is useful when large input values are to be put into a field.

`getval` *<name>*: Get the value of the field named by *attrname.*

`new`: Create a new empty tree representation of an IODEF object.

`quit`: quit the IHSH command interpreter.

`readxml` *<file>*: Read XML formatted report from file *file* and reprent the contents internally as a tree.

`setval` *<name>* (*val*): At the current location in the tree, set attribute *name* to the value of *val.* If *name* is a full path, it is assumed to name an attribute globally in the tree. If attrval is omitted, input is assumed to be of multi-line format and must be terminated by a final line containing only a dot (.).

`setattr` *<attr>* (*val*): At the current location in the tree, set attibute *attr* to value *val.*

`writexml` *<file>*: Write XML formatted report of the internal IODEF tree representation.

3.2 Perl Library

3.2.1 Overview

IODEF.pm is Perl interface that can be used to create and parse IODEF messages. It is compliant with IODEF v1.0. The interface has been designed for simplifying the task of translating a key-value based format to its iodef representation. A typical session involves the creation of a new IODEF message, the initialization of some of it's fields and its conversion into an IODEF string.

This module is based on *XML::DOM* and contains a simplified version of the latest IODEF DTD. It is hence DTD aware and perform some validity checks on the IODEF message treated, in an attempt at easying the process of producing valid IODEF messages. The simplified internal DTD representation can easily be upgraded or extended to support new XML node.

3.2.2 Functions

Below follows a summary of the functions provided through the *IODEF.pm* interface:

new: create new IODEF message

in: load new IODEF message from string/file

out: write IODEF message to string/file

to_hash: convert IODEF message to hash for easy parsing

add: add a field to IODEF message

get_type: return type of IODEF message

create_ident: initialyze the Incidentident field witha unique id string

create_time: initialize the CreateTime field with the current time

4 Status and Concluding Remarks

During the project period all CSIRTs participating in the project have integrated the common language into experimental CSIRT operation using one of the above mentioned tools. Due to the simplicity and ease of prototype implementation most teams have used the IODEF.pm Perl library. One of the teams have used IHSH to integrate with the ARS/Remedy workflow management system. Concluding, at the end of the project all participating CSIRTs have a method integrated into (experimental) operation that enables them so send and receive incident information according to the eCSIRT.net common language.

References

[Koek et al., 2001] Koek, M., Smits, E., Stikvoort, D., and Kossakowski, K.-P. (2001). The Trusted Introducer Service. In *FIRST Conference on Computer Security Incident Handling & Response 2001*, Toulouse, France.

[West-Brown et al., 1998] West-Brown, M. J., Stikvoort, D., and Kossakowski, K.-P. (1998). Handbook for Computer Security Incident Response Teams (CSIRTs). Technical Report CMU/SEI-98-HB-001, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA.