

A Survey on Languages, Enumerations and Other Tools used for Security Information Communication and Sharing

ANTEL
TELECOMUNICACIONES



LACNIC XI
26/30 MAIO 2008 SALVADOR/BAHIA - BRASIL

CSIRT
GESTIÓN DE INCIDENTES DE SEGURIDAD
INFORMÁTICA Y DE TELECOMUNICACIONES



GRUPO DE SEGURIDAD INFORMÁTICA

LACNIC XI - Salvador

May 28th, 2008

Presented by Carlos M. Martínez-Cagnazzo

carlos.martinez@csirt-antel.com.uy

Agenda

- Intro to CSIRTs: Computer Security Incident Response Teams
- Approaches for Security Data & Information Standardization
- Enumerations
 - CVE
- Languages
 - CVE, OVAL
- Conclusions & Final Remarks
- About the authors
- References



INTRO TO CSIRTS

What is a CSIRT ?

- **CSIRT**: Computer Security Incident Response Team
- As defined in [CERT00], a Computer Security Incident Response Team is a service organization which provides a clearly-defined set of services to an also clearly-defined constituency
- A CSIRT's ultimate goal is to improve its constituency's preparedness and response capabilities in the face of computer security incidents. Its services must be tailored to the constituency's needs and idiosyncrasies

Some Definitions

- **Incident**

There is no unanimously-agreed definition of what an incident is. We consider the definitions from RFC 2828 and RFC4949 to be adequate for our purposes.

In their words, a security incident is a security event in which the system's security policy is disobeyed or otherwise breached

- **Constituency**

Constituency is the company, university, group of companies or country that the CSIRT serves

- **Services**

Services can be broadly classified as reactive, proactive and value-added services. There is no standard set of services and each team must build their service offerings according to their host organizations' needs and the team's own capabilities.

Services At A Glance

- Source: <http://www.cert.org/csirts/services.html>

Reactive Services	Proactive Services	Security Quality Management Services
Alerts and Warnings	Announcements	Risk Analysis
Incident Handling <ul style="list-style-type: none">- Incident analysis- Incident response on site- Incident response support- Incident response coordination	Technology Watch	Business Continuity and Disaster Recovery Planning
Vulnerability Handling <ul style="list-style-type: none">- Vulnerability analysis- Vulnerability response- Vulnerability response coordination	Security Audits or Assessments	Security Consulting
Artifact Handling <ul style="list-style-type: none">- Artifact analysis- Artifact response- Artifact response coordination	Configuration and Maintenance of Security Tools, Applications, and Infrastructures	Awareness Building
	Development of Security Tools	Education/Training
	Intrusion Detection Services	Product Evaluation or Certification
	Security-Related Information Dissemination	

Communicational Needs of a CSIRT

- Effective communication is key to a CSIRT's success. While providing services, CSIRTs receive, generate and process security-related data and information.
- As noted by [9], the domain of computer security is far from being fully standardized. In many cases terminology is not uniform and heavily dependent on context.
- It is possible to name some broad data categories, for example:
 - Network traffic events
 - Incident reports
 - Vulnerabilities
 - Artifacts

Communicational Needs of a CSIRT

(II)

- Information is exchanged between a CSIRT and its constituency or between a CSIRT with other CSIRTs.
- These exchanges allow the teams to:
 - Request additional information from entities reporting possible incidents
 - Receive activity reports from other teams in order to build and maintain *situational awareness*
 - Receive vulnerability reports from other teams
 - Provide other teams with similar information

The Need for Standardization

- **Normalization**

A team processing information has to invest resources into normalizing the information it receives or either risk duplicating efforts or overlooking some important bit of data.

There have been numerous attempts to create naming schemes and other tools to express security-related information.

However there has been a lot of disagreement within the Security community and these approaches have led to the same malware or vulnerability to be named very differently under different schemes.

- **Trust**





A CSIRT needs to be able to evaluate how trusted a certain set of data is in order to make informed decisions or to commit resources to evaluate a certain threat.



APPROACHES FOR SECURITY DATA STANDARDIZATION

Taxonomy

- There are numerous approaches that have been or are in use. We find useful the classification found in [MITRE00] that distinguishes between *Enumerations, Languages and Repositories*

Enumerations	Languages	Repositories
 <u>Common Vulnerabilities and Exposures (CVE®)</u> - common vulnerability identifiers	 <u>Open Vulnerability and Assessment Language (OVAL®)</u> - standard for determining vulnerability and configuration issues	 <u>OVAL Repository</u> - community-developed OVAL Vulnerability, Compliance, Inventory, and Patch Definitions
 <u>Common Weakness Enumeration (CWE™)</u> - list of software weakness types	 <u>Common Result Format (CRF™)</u> - standardized assessment result format for conveying findings based on common names and naming schemes	<u>National Vulnerability Database (NVD)</u> - U.S. vulnerability database based on CVE that integrates all publicly available vulnerability resources and references
 <u>Common Attack Pattern Enumeration and Classification (CAPEC™)</u> - list of common attack patterns	 <u>Common Event Expression (CEE™)</u> - standardizes the way computer events are described, logged, and exchanged	<u>NIST Security Content Automation Protocol (SCAP)</u> - security content for automating technical control compliance activities, vulnerability checking, and security measurement
 <u>Common Malware Enumeration (CME™)</u> - common identifiers for viruses, worms, and other malicious code	<u>OVAL Interpreter</u> - free tool for collecting	<u>Red Hat Repository</u> - OVAL Patch

<http://measurablesecurity.mitre.org/>

Enumerations

- **Enumerations** are basically *lists of items*, defined and maintained by a certain central authority
- These items can be for example vulnerabilities, malware, platforms or application flaws
- Examples of enumerations include:
 - CVE: Common Vulnerabilities and Exposures
 - CME: Common Malware Enumeration

Languages

- Languages are data models to express security-related information. All the current ones are XML-based and all try to go beyond what enumerations do in terms of normalization
- Examples of languages include:
 - OVAL: Open Vulnerability and Assessment Language
 - IODEF: Incident Object Definition Exchange Format

Repositories

- Repositories are universally-recognized authorities that store security-related information
- This information could use an enumeration-based or a language-based approach, or even both, since neither system tries to enforce local storage methods
- Repositories serve two main purposes:
 - They provide single points of concentration and storage of information, where the community can refer for search and data retrieval
 - They provide an implicit trust model, since we as a community recognize the repository owner's authority over the repository's database and implicitly trust him to ensure the db's integrity and authenticity

Examples of Repositories

- Examples of repositories include:
 - NVD: National Vulnerability Database, a CVE-based vulnerability repository
 - Mitre's OVAL repository
 - RedHat patch definition repository, based on OVAL



ENUMERATION EXAMPLE: CVE

Common vulnerabilities and Exposures

- CVE was created and developed by the MITRE Corp., See [6] and <http://cve.mitre.org>
- CVE aims for normalizing vulnerability counts and reducing the confusion that surrounds the announcement and counting of different vulnerabilities and to allow for interoperation of different tools

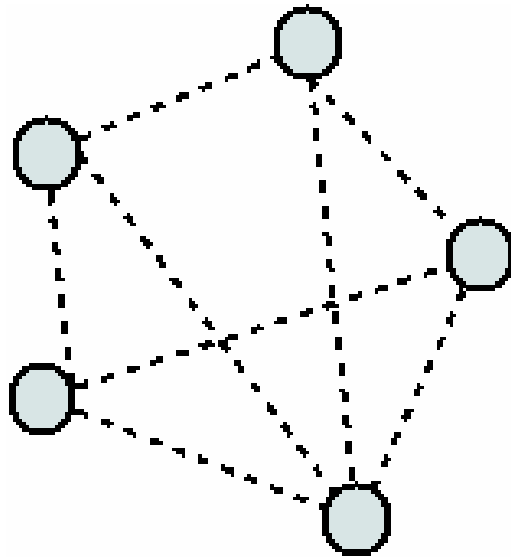


CVE (2)

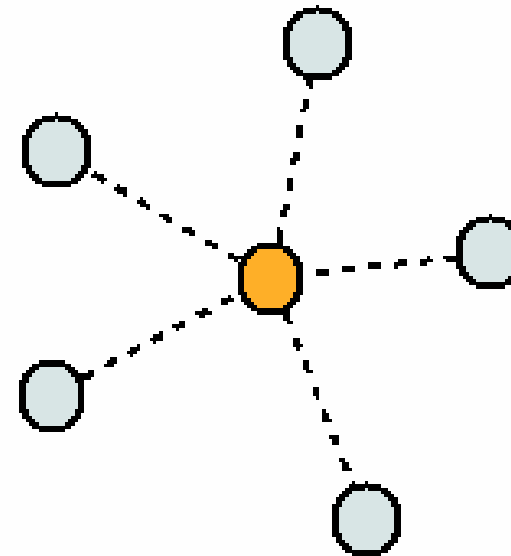
- [MAN99] mentions the following goals driving the development of CVE:
 - It must enumerate and discriminate between all known vulnerabilities
 - It must assign a standard, unique name to each vulnerability
 - It exists independently of the multiple perspectives of what a vulnerability is
 - Is publicly "open" and shareable without restrictions
- The authors also refer to some secondary goals, including:
 - Do not attempt to enumerate characteristics or attributes, but rather have few required fields and many free form descriptive fields
 - Serve as a *logical bridge* across other naming schemes and databases

CVE (3)

- CVE as a *logical bridge* according to [MAN99]



Without CVE



With CVE

CVE Creation Process

- Entries to the CVE database are added according to a three-stage process that includes:
 - Initial submission
 - Candidate stage
 - Entry stage
- This process is controlled by an Editorial Board that assigns CVE IDs according to information gathered from different sources and the opinions of a team of qualified security researchers
- The Editorial Board is responsible for accepting entries in a fashion that respects the enumeration's design goals

Example of a CVE Entry

Name: CVE-2002-0148

Description: Cross-site scripting vulnerability in Internet Information Server (IIS) 4.0, 5.0 and 5.1 allows remote attackers to execute arbitrary script as other users via an HTTP error page. **Status:** Entry

Reference: BUGTRAQ:20020410 IIS allows universal CrossSiteScripting

Reference: MS:MS02-018

Reference: URL:http:

[//www.microsoft.com/technet/security/bulletin/ms02-018.asp](http://www.microsoft.com/technet/security/bulletin/ms02-018.asp)

CVE Repositories

- MITRE CVE Master List

MITRE Corp. (CVE's creator) maintains a master list of created CVE's. As of May 12, 2008 the list contains 30741 entries

- NVD: *National Vulnerability Database*

According to [MITRE00] the NVD is a CVE-based database integrating all currently known vulnerabilities and references.



LANGUAGES

IODEF

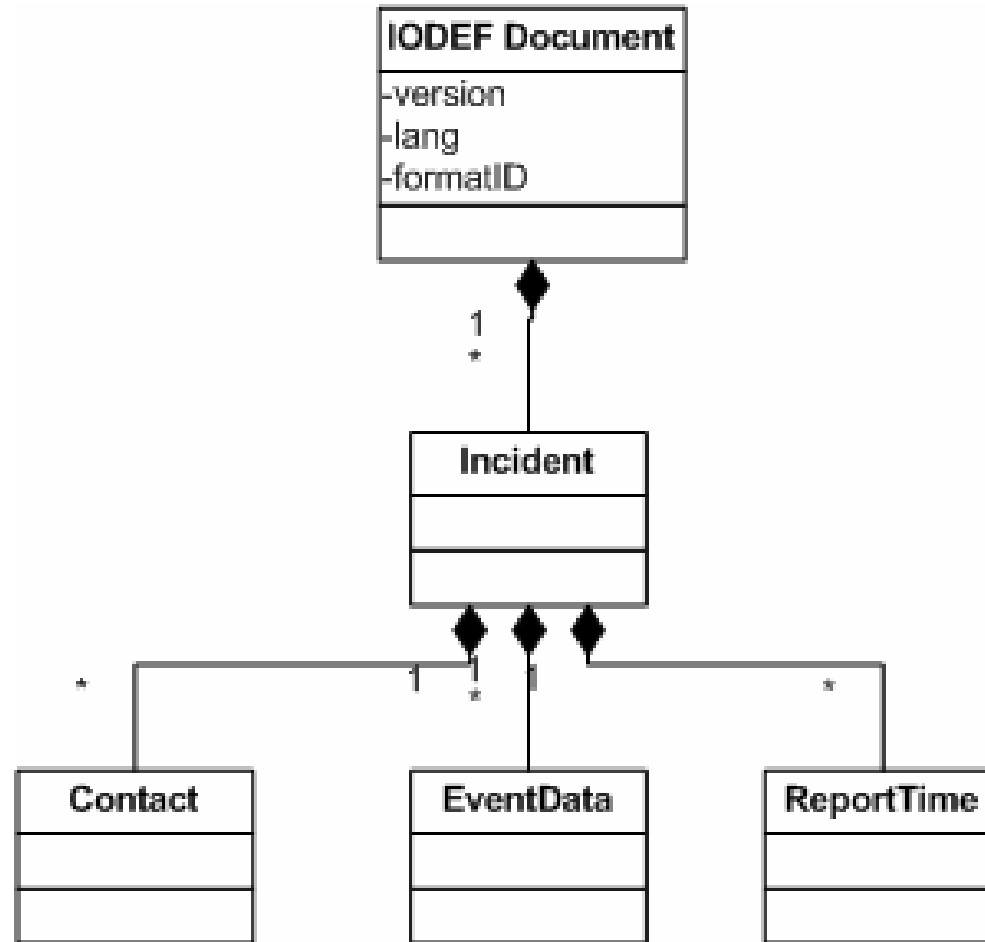
- The Incident Object Denition Exchange Format is an XML-based language created by the IETF and documented in RFC 5070 ([7])
- The original work was initiated by TERENA and latter submitted to the IETF, which continued the standardization process
- Other previous work includes IDMEF, the *Intrusion Detection Message Exchange Format*, documented in RFC 4765 ([8]).

IODEF's Design Goals

- According to RFC 5070 [7], IODEF is a format that is used to represent computer security information exchanged between CSIRTs.
- Its main goal is to enhance CSIRT's operational capabilities by providing teams with normalized, automatically process able information
- Specific goals include:
 - Increase automation in processing incident data, by standardizing many fields and data objects
 - Decreased effort in normalizing similar data
 - A common format on which to build interoperable tools for incident handling and subsequent analysis

IODEF's Schema Class Diagram

- Main classes



Trust in IODEF

- IODEF does not define nor mandate a trust model and chooses rather to specialize on expressing security information
- RFC 5070 explicitly acknowledges that the properties of confidentiality, integrity and authenticity must be assured by the underlying transport mechanism or storage method
- As of May, 2008 there is an (recently renewed) IETF draft (draft-moriarty-post-inch-rid) that defines a SOAP transport scheme for IODEF

IODEF Example from RFC 5070

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- This example demonstrates a report for a very old worm (Code Red) -->
<IODEF-Document version="1.00" lang="en"
  xmlns="urn:ietf:params:xml:ns:iodef-1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="urn:ietf:params:xml:schema:iodef-1.0">
  <Incident purpose="reporting">
    <IncidentID name="csirt.example.com">189493</IncidentID>
    <ReportTime>2001-09-13T23:19:24+00:00</ReportTime>
    <Description>Host sending out Code Red probes</Description>
    <!-- An administrative privilege was attempted, but failed -->
    <Assessment>
      <Impact completion="failed" type="admin"/>
    </Assessment>
    <Contact role="creator" type="organization">
      <ContactName>Example.com CSIRT</ContactName>
      <RegistryHandle registry="arin">example-com</RegistryHandle>
      <Email>contact@csirt.example.com</Email>
    </Contact>
  </Incident>
</IODEF-Document>
```

OVAL

- **OVAL**: Open Vulnerability and Assessment Language
- According to [OVAL02], the OVAL language strives for standardizing the transfer of security information across the whole spectrum of security tools and services
- OVAL standardizes the three main aspects of the assessment process:
 - Representing configuration information
 - Analyzing a system for the presence of a specified state (vulnerability, configuration, patch level)
 - Reporting the results of the assessment process

OVAL Use Cases

- [5] mentions a number of language use cases that illustrate some of OVAL's goals:
 - Security Advisory Distribution
 - Provide vulnerability information in a standard, machine-readable form.
 - Vulnerability Assessment
 - Once a vulnerability is disclosed, organizations must engage in a labor-intensive testing process. One use of OVAL is expressing series of tests that can be used to assess many systems for the presence of a vulnerability.
 - Patch Management
 - Applying patches is not a straightforward process, involving sometimes a certain amount of reverse engineering. OVAL can also be used to express the required state of a system so an updating tool is able to apply a certain patch.

OVAL's XML Schemas

- In order to address the complexity of all language use cases, OVAL's XML schema has been defined in a modular way
- OVAL Definition Schema
 - The Definition Schema is used to write (1) vulnerability definitions, (2) patch definitions and (3) compliance definitions
- OVAL System Characteristics Schema
 - The System Characteristics Schema is used to represent system configuration information. This includes for example OS parameters, installed software, applications settings
- OVAL Results Schema
 - The Results Schema is used to represent the result of a system assessment process

OVAL: Core and Component Schemas

- All of OVAL's schemas are not monolithic but rather composed of a Core schema plus one or more Component schemas
- Core schemas define core properties
- Component schemas are created in order to isolate the specifics of different platforms and operating systems. Examples:
 - Debian, RedHat, Solaris component schemas
- OVAL definitions can use tests pulled from the different schemas
 - Each vendor can thus create and maintain tests for its platforms without the need of modifying existing schemas.

Trust in OVAL

- As with IODEF, OVAL does not define an explicit trust model
- There are at least two publicly accessible OVAL repositories from where definitions can be retrieved
- Repositories:
 - MITRE's OVAL Repository
 - RedHat's OVAL Repository



CONCLUSIONS & FINAL REMARKS

Conclusions

- Security information hard to standardize, formalize
- Different approaches, different goals
 - Transport
 - Logical bridges
 - Data models
- Trust models: there is still a lot to do
 - Currently we trust repositories, but we perceive the need for a trust model that enables peer-to-peer interaction
- Currently two main efforts underway, IETF and MITRE
- Enumerations well-established, CVE for example



ABOUT THE AUTHORS

The Authors

- This work is part of a wider research project on security information transport, handling and processing
- CSIRT-Antel:
 - Eduardo Carozo, CSO ANTEL
 - Carlos M. Martinez-Cagnazzo
- G.S.I. - Facultad de Ingeniería
 - Gustavo Betarte
 - Marcelo Rodriguez
 - Alejandro Blanco

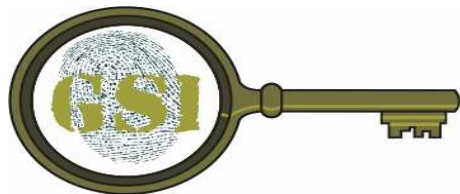
CSIRT-ANTEL

- Created on December 2005 by ANTEL (Administración Nacional de Telecomunicaciones) in Uruguay
- Member of FIRST since May, 2007
- Responded to more than 200 incidents since creation, including:
 - Denial of Service Denial Attacks
 - Information disclosure incidents
 - Attacks against the company's websites



G.S.I. – Universidad de la República

- Founded beginning 2006
- Team members come from Computer Science and Electrical Engineering departments
- Activities
 - Education (undergraduate and graduate programs)
 - Research
 - Development of R&D projects with industrial partners
- Research areas
 - Development of methodologies and infrastructures for secure distributed computations
 - Model and languages for decentralized authorization
 - Development of models and tools for the analysis and management of computer security incidents
 - Formal specification and Verification of Embedded Systems
 - Digital Forensics



GRUPO DE SEGURIDAD INFORMÁTICA





REFERENCES

References

- [CERT00] “*Computer Security Incident Response Teams FAQ*”
(http://www.cert.org/csirts/csirt_faq.html)
- [MITRE00] “*Making Security Measurable*” website
(<http://measurablesecurity.mitre.org/>)
- [MANN99] “*Towards a Common Enumeration of Vulnerabilities*”, D. Mann, S. Christey, 1999
- [OVAL00] “*An Introduction to the OVAL Language*”, MITRE Corp.,
(http://oval.mitre.org/oval/documents/docs-06/an_introduction_to_the_oval_language.pdf)
- [OVAL02] “*About OVAL*” website
(<http://oval.mitre.org/oval/about/index.html>)

References (2)

- [RFC2828] “*Internet Security Glossary*”, R. Shirey
(<http://www.ietf.org/rfc/rfc2828.txt>)
- [RFC4949] “*Internet Security Glossary, Version 2*”
(<http://www.ietf.org/rfc/rfc4949.txt>)



THANK YOU VERY MUCH!