

International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1206

(04/2008)

SERIES X: DATA NETWORKS, OPEN SYSTEM
COMMUNICATIONS AND SECURITY

Telecommunication security

**A vendor-neutral framework for automatic
notification of security related information and
dissemination of updates**

Recommendation ITU-T X.1206

ITU-T X-SERIES RECOMMENDATIONS
DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	
Services and facilities	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalling and switching	X.50–X.89
Network aspects	X.90–X.149
Maintenance	X.150–X.179
Administrative arrangements	X.180–X.199
OPEN SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	X.220–X.229
Connectionless-mode protocol specifications	X.230–X.239
PICS proformas	X.240–X.259
Protocol Identification	X.260–X.269
Security Protocols	X.270–X.279
Layer Managed Objects	X.280–X.289
Conformance testing	X.290–X.299
INTERWORKING BETWEEN NETWORKS	
General	X.300–X.349
Satellite data transmission systems	X.350–X.369
IP-based networks	X.370–X.379
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	
Networking	X.600–X.629
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	X.680–X.699
OSI MANAGEMENT	
Systems Management framework and architecture	X.700–X.709
Management Communication Service and Protocol	X.710–X.719
Structure of Management Information	X.720–X.729
Management functions and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
Commitment, Concurrency and Recovery	X.850–X.859
Transaction processing	X.860–X.879
Remote operations	X.880–X.889
Generic applications of ASN.1	X.890–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
TELECOMMUNICATION SECURITY	X.1000–

For further details, please refer to the list of ITU-T Recommendations.

Recommendation ITU-T X.1206

A vendor-neutral framework for automatic notification of security related information and dissemination of updates

Summary

Recommendation ITU-T X.1206 provides a framework for automatic notification of security related information and dissemination of updates. The key point of the framework is that it is a vendor-neutral framework. Once an Asset is registered, updates on vulnerabilities information and patches or updates can be automatically made available to the users or directly to applications regarding the Asset.

Source

Recommendation ITU-T X.1206 was approved on 18 April 2008 by ITU-T Study Group 17 (2005-2008) under the WTSA Resolution 1 procedure.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at <http://www.itu.int/ITU-T/ipr/>.

© ITU 2009

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.

CONTENTS

Page

1	Scope	1
2	References.....	1
3	Definitions	1
	3.1 Terms defined in this Recommendation.....	1
4	Abbreviations.....	2
5	Conventions	2
6	Introduction	2
7	Current situation regarding vulnerability information.....	3
8	Overview of vendor-neutral framework	5
	8.1 Multiple sources of vulnerability information, updates and patches.....	5
	8.2 Example application operation	6
	8.3 Security and privacy considerations	7
9	Recommendation architecture	7
	9.1 Message core layer	7
	9.2 Message/application layer	8
	9.3 Scalability	8
	9.4 Extensibility.....	8
	9.5 Platform independence	9
	9.6 Client/Server communication.....	9
10	Components of the framework	9
	10.1 Message container	9
	10.2 Version message	13
11	Schemas	19
	11.1 Message_Core	19
	11.2 Message_Version	21
	Bibliography.....	26

Recommendation ITU-T X.1206

A vendor-neutral framework for automatic notification of security related information and dissemination of updates

1 Scope

This Recommendation provides a framework of bidirectional flow of automatic notification and distribution of vulnerability information as well as the distribution of updates and/or patches. In addition, this Recommendation makes it possible for system administrators to know the condition of any Asset within their realm of responsibility.

Clauses 6 and 7 describe the problems of maintaining Assets from an Asset identification point of view, as well as information dissemination and systems/networks management points of view.

Clause 8 describes the overview of the vendor-neutral framework, which includes an example system supported by the adoption of the framework, compartments of the framework and an exemplary sequence of exchanges within the framework. Clause 8 also describes the security that should be considered in the vendor-neutral framework.

Clause 9 describes the functionalities and features of this Recommendation.

Clause 10 provides the definitions of the data structures of components of this Recommendation.

Clause 11 contains the XML schema defined and described in clause 10.

This Recommendation provides a framework that any vendor can use for notification, as well as the receiving of vulnerability information and dissemination of required patches/updates for covered Assets, and defines the format of the information that should be used in and between components implementing this framework.

This Recommendation does not define protocols to be used in the communication between components as many protocols are supported without special consideration.

Some common roles and responsibilities will be needed to be established for operation based on the vendor-neutral framework; however, a discussion regarding the establishment and operation of possible roles and their resulting responsibilities is not within the scope of this Recommendation.

2 References

None.

3 Definitions

3.1 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.1.1 agent: An implementation of this Recommendation operating in support of an installed asset on a given device, in support of server functionality or in support of local server functionality.

3.1.2 asset: A device, separately identifiable piece of hardware, application, operating system or instance of executable code.

3.1.3 client: A device which requests services from another device.

3.1.4 device: A system acting as either a client, server, or both, local server.

3.1.5 group: A number of devices operated on as a single unit.

3.1.6 local server: A client acting as a server node for additional downstream clients.

3.1.7 message: A request for a specific action to be performed, e.g., general actions such as "Register" an asset as being of a given version and/or contained components of given versions, "Request" existing or future available updates, patches or vulnerability information, etc. Messages extending the functionality of this Recommendation may be defined outside the scope of this Recommendation.

3.1.8 message data: Information provided in support of a given message. Among an almost infinite number of possibilities, specific examples defined in this Recommendation are data defining version information, vulnerability information pertaining to given versions as well as updates or patches for specific versions.

3.1.9 message set: A combination and association of a universally unique identifier, a message and the message's associated message data's definition, all defined within an XML schema derived from and extending *Message_Core* defined herein.

3.1.10 patch: A broadly released fix for a product-specific, security-related vulnerability. A method of updating a file that replaces only the parts being changed, rather than the entire file.

3.1.11 server: A device used to service requests from other devices.

3.1.12 vulnerability: Any weakness, administrative process or act, or physical exposure that makes a computer or network of computers susceptible to exploit by a threat.

4 Abbreviations

This Recommendation uses the following abbreviations and acronyms:

API	Application Programming Interface
GUID	Globally Unique IDentifier
HTTP	HyperText Transfer Protocol
ISIRT	Information Security Incident Response Team
ISP	Internet Service provider
OS	Operating System
POAS	Platform/Operating System/Application/Service
URI	Uniform Resource Identifier

5 Conventions

None.

6 Introduction

As more people begin to use computers in their homes and workplaces, and fewer have any kind of official training in the operation of their computers, let alone security-related issues, one quickly approaches the point where security, not only becomes almost impossible to maintain, but it becomes more difficult for those responsible for maintaining security on a system level to know much about the condition of the systems they are responsible for and provide services until some breach or accident occurs, by which time it is already too late.

That is primarily due to the fact that so many different computers are in different states of maintenance and update. Where security-related issues are concerned, system management is much less of a preventative process than of a disaster management and recovery process.

Although a number of applications and even operating systems (OS) have their own update mechanisms, they all have a number of problems in common. One such problem is that all the update mechanisms rely on being enabled, in the first place, and, in the second, allowed to do their job when the user is notified of an update being available, assuming that the user has enabled the notifications.

Likely worst of all though is that these problems leave system administrators totally out of the picture, so that without installing their own monitoring systems on each computer under their responsibility, they have no idea as to the general level of security within the networks and systems they are responsible for.

Another consideration is that while updating software to the latest available it is often the case where updates alone are not the solution, but instead improved usage practices for which no update, other than the information being received by the end user, is of use. Even though various applications and OSs may have updating mechanisms in place, none of them has a uniform method of keeping users informed of the latest best practices leading to continued secure use.

Also of importance are the methods used to distribute updates. Currently, all updates through the various update mechanisms source the updates through dedicated channels, one for each update session. But where updates, or other important information made available to various redistribution centers, e.g., ISPs or corporate networks, are subsequently distributed within the networks in a trusted and secure manner, the bandwidth required for distribution could effectively be cut in half or at least reduced to a great extent just as in the case of hypertext transfer protocol (HTTP) proxies.

Another concern that is mismanaged in most cases is where users themselves find a problem regarding the use or actions of a given asset, without having anyone to refer the problem to. Even if some method is in place for users to contact system administrators or other support personnel, using such a system ends up more like a game of "20 questions" where the user and support person must go back and forth asking for information and replying usually with more questions.

It is often difficult for even an experienced support person to have a clear understanding of what exactly comprises many assets. Due to modular software architectures and various pieces of software actually comprising modules from different vendors, all with their own versioning systems, even if an update or relevant piece of information about a given product or sub-module is made available, knowing to what and to whom it applies can be a difficult task, often leading the less informed to ignore much of what their systems may rely for security.

In the end, one ends up with users being uninformed and those responsible for system wide security being essentially left out of the loop.

7 Current situation regarding vulnerability information

Vulnerability information is presently released by many vendors and many security-related organizations, such as the Information Security Incident Response Team (ISIRT), in an effort to make users aware of the security-related issues as well as providing updates and patches, when required. However, it is often the case that end users neither make use of the information, updates or patches or even know if, whatever is provided, applies to them.

There are various reasons for this situation, but first one should understand why end users might find it hard to make use of the information made available to them.

An end user normally uses version information of assets to determine whether they include the system, software or components that are affected by a newly-founded vulnerability. However, there are some difficulties for using version information to detect the affected system, software or components.

- Ambiguity of version notation

The rules of version notation vary from vendor to vendor. This variation may cause the end user's interpretation of the version information to be different from the intention of the vendor. Not understanding the vendor-specific version system, end users may not know that a given update, patch or information applies to them.

For example, in the case that there are 3 versions, "v1", "v1.1", and "v1.2", of the system that are described as a target in the vulnerability information, if the vulnerability information is described as below.

"Systems Affected

* XXXX v1 the affected system "

Could have the following two different meanings:

- 1) v1 is just v1.0.
- 2) v1 means v1.x, so all sub-versions of version 1.

Trying to standardize the version notation is not a practical solution; so it is necessary to read and understand the vendor-specific rules of the version notation, but it is not realistic to hope that end users will do so.

- The version of a given product cannot always be used as a criteria for determining what may actually be vulnerable.

For some vendors, the version of the products that are generally known by the users might not be able to be used to detect whether the product has vulnerabilities or not. (Figure 1)

In most component-oriented products, the presence of a vulnerability may have to be determined by the version of each component that comprises the product, not the version of the product itself.

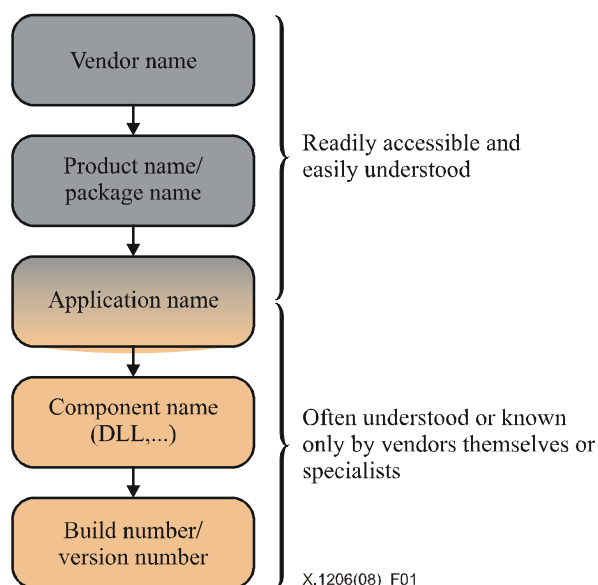


Figure 1 – Hierarchic structure of "version"

- Products that embed another vendor's products as components

Most vulnerability information usually refers to a given product. An end user may not use the specific product referred to, but instead may use the product through the use of another application that loads or otherwise makes use of a given vulnerable product's service. As a matter of fact, with open published application programming interfaces (APIs), a vendor of products used in other applications may have little to no knowledge of where their products are used or in what other applications they may be loaded or otherwise used.

- System administrators are unaware of the condition of assets under their responsibility

As system administrators are dependent on users maintaining their own systems and without personally verifying the status of each and every system themselves or relying on self reporting, it is impossible to know the status of the systems they are ultimately responsible for. Without self reporting, which is often inaccurate and incomplete for many of the reasons already mentioned regarding versioning or, custom monitoring applications, which require development and maintenance, those responsible for maintaining the security of corporate or Internet service provider (ISP) networks have few tools available.

8 Overview of vendor-neutral framework

This clause describes the overview of the vendor-neutral framework for the distribution of vulnerability, update and patch information.

By adopting the framework, vulnerability information, update and patch distribution systems, such as is shown in Figure 2, may be constructed using a simple subscription scheme. Each asset, device or local server may register with any and all available servers to receive requested (pull) or suggested (push) vulnerability information, updates and/or patches.

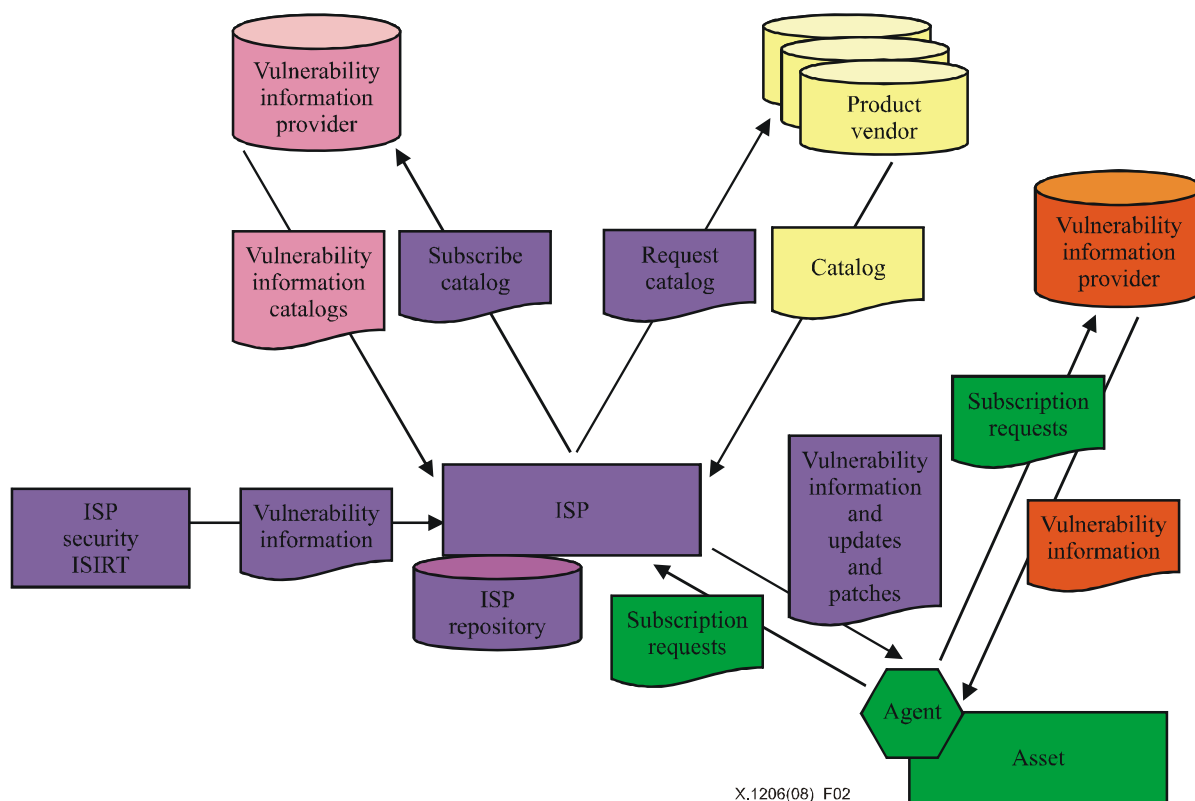


Figure 2 – Example application architecture

8.1 Multiple sources of vulnerability information, updates and patches

Using the same message scheme, any entity may make requests and/or provide vulnerability information, updates and/or patches to any other entity. In this specific example, an ISP is acting as a collection point to act as a source to all its subscribers. Additionally, individual subscribers can register with independent third-party sources for information updates, code analysis or even code updates or patches. Of course, it would be necessary for any application attempting to apply a given update or patch to accept the update or patch as authorized, but how this is accomplished, other than somehow using the supported inclusion of signatures, is beyond the scope of this Recommendation.