A Model to Reduce IDS Alerts

MARCO AURÉLIO BONATO, WALTER GODOY JR.
Centro de Pós-Graduação em Engenharia Elétrica e Informática Industrial
Centro Federal de Educação Tecnológica do Paraná
Av. Sete de Setembro, 3165 Cep: 80.230-901 – Curitiba - Paraná
BRASIL

Abstract: - The goal of this paper is to present a model to reduce the alerts generated by an IDS[01] analyzer. This model allows the administrator to analyze only the messages that really generate risks for an environment or machine. This is very important when you have a complex environment with a lot of machines with many services in them.

Key-Words: - Network Security, Intrusion Detection, IDS.

1 Introduction

One of the biggest problems in IDS administration is the generation of false positive alerts or alerts that do not bring risks to an environment or machine. One false positive alert occur when normal packages are identified as attacks. In a complex environment of intrusion detention, a large number of alerts of this kind can harm the analysis of true attacks.

The configuration of any type of analyzer is a complicated task because it demands from the administrator deep knowledge in protocols, applications and attacks format. However, if the administrator is able to make this administration through a friendly platform, the analyzer can detect all the alerts, the filtering being made in the configuration of the profile of each network/machine. This platform is represented as an object oriented model (UML representation)[02] and has three parts. The first part show one model to represent the environment or machine, the second part show the way that the attacks database must be organized to facilitate the analysis, and the third part is the program that analyzes the alerts. This program uses the environment definition plus the attacks database to define if the alerts bring risks to the environment or machine.

2 Environment

The Environment class (Fig. 1) defines the environment where the analyzer is installed. Through this class and its aggregations it is possible to create a profile for each computer or network.

The Environment class has four attributes:

- description: optional. One brief description about the environment;
- location: optional. The location of the environment;
- address: required. The environment network address;
- netmask: optional. The network mask for the address.

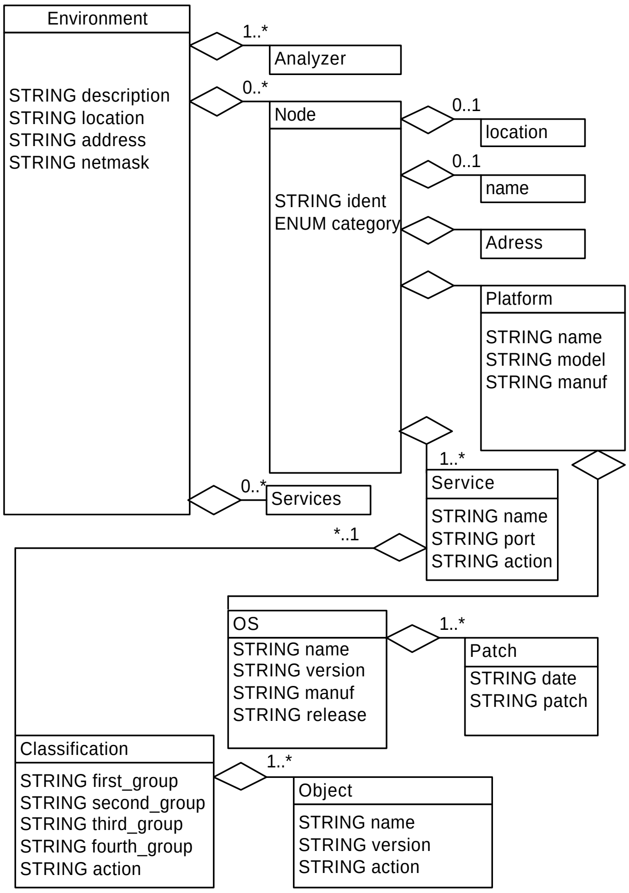


Fig. 1 The Environment Model

2.1 The Analyzer class

The Analyzer class identifies the analyzer that is installed in the environment. This class is defined in [03].

2.2 The Node class

The Node class is used to identify hosts and others devices. In this case it is used to identify the hosts that belong to an environment.

The Node class has two attributes:

- ident: optional. A unique identifier for the node;
- category: optional. The "domain" from which the name information was obtained, if relevant. The permitted values for this attribute are showed in [03]. The default value is "unknown".

The Node class has two simple aggregated classes that are defined here:

- location: zero or one. STRING. The location of the equipment;
- name: zero or one STRING. The name of the equipment.

2.3 The Address class

The Address class is used to represent the network address for a node. This class is defined in [03].

2.4 The Platform class

The Platform class is used to define the node characteristics. This is useful because some attacks are targeted only to some specific platforms.

The Platform class has three attributes:

- name: required. The platform name. Examples: Intel, Hp, Sun, etc.;
- model: optional. The platform model;
- manufacturer: optional. The manufacturer of the platform.

2.5 The OS class

The OS class is used to define the operational system installed in the platform. This is useful because some attacks are targeted only to some specific operational systems.

The OS class has three attributes:

- name: required. The OS name. Examples: Unix, Linux, Windows, etc.;
- version: required. The OS version;
- manufacturer: optional. The manufacturer of the OS;
- release: required. The current operational system release.

2.6 The Patch class

The Patch class is used to define the patch that has been applied to the operational system. This is useful because some attacks are targeted only to some specific operational systems versions. Some patches correct vulnerabilities, so it's important to know this information.

The Patch class has two attributes:

- date: required. The date on which the patch was applied to the operational system;
- patch: required. What is the patch that was applied (version, name, number, etc);

2.7 The Service class

The Service class is used to define the service that is being executed. This service can be associated to an environment or to a single machine.

The Service class has three attributes:

- name: required. The service name. Examples: Web, Ftp, Telnet, etc;
- port: required. The tcp/udp port which is being used by the service;
- action: required. What to do with the alerts that are being received by this machine/service, "accept" or "reject";

2.8 The Classification class

The Classification class is used to define the service classification. This is used to detail the service.

The Classification class has five attributes:

- first_group: required. The first classification. Examples: for the service Web we can use Apache, IIS, Netscape, etc. The service that does not have a specific classification receive "general" for this attribute;
- second_group: optional. The second classification, if necessary. Examples: For the service Web, first classification IIS we can use FrontPage as a second classification;
- third_group: optional. The third classification, if necessary;
- fourth_group: optional. The fourth classification, if necessary;
- action: required. What to do with the alerts that are being received by this machine/service/classification, "accept" or "reject";

2.9 The Object class

The Object class is used to define the service object.

The Object class has three attributes:

- name: required. The object name. Examples: for the service Web we can use Apache for the first classification and we can define the object httpd;
- version: required. The current object version;
- action: required. What to do with the alerts that are being received by this machine/service/classification/object, “accept” or “reject”;

3 Attack

The Attack class (Fig. 2) identifies and organizes the atack information. With the environemnet information (Environment class) and the atack information (Atack class), it becomes more easy to administer and minimize alerts. The environment caracteristics are compared with the attack database to verify if the alert brings risks to the environment or machine.

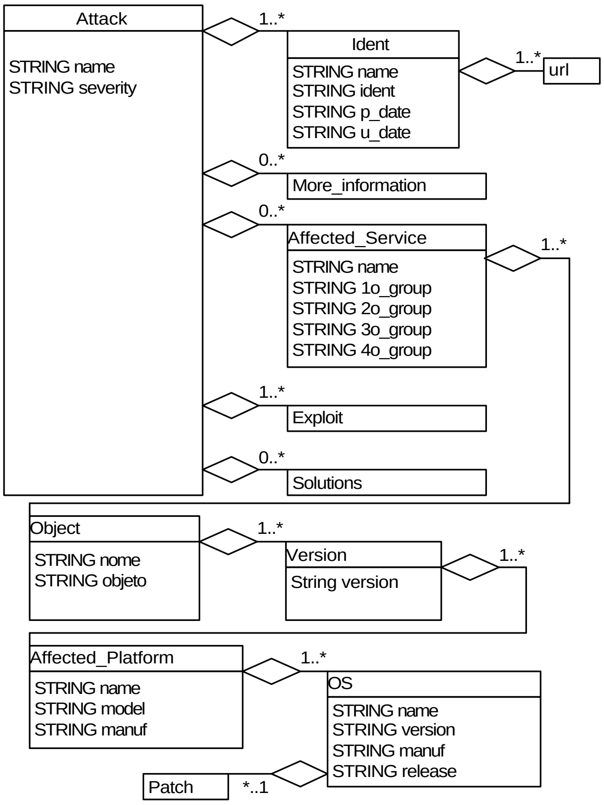


Fig. 2 The Attack Class

The Attack has two attributes:

- name: required. A generic name to the attack;
- severity: optional. The risks that the attack can cause to the environment or machine. The permitted value for this attribute are shown bellow;

Rank	Description
0	Low risk
1	Medium risk
2	High risk

Table 1 Severity

3.1 The Ident class

The Ident class is used to determine for each unique attack its relationship with all databases notification attacks available in the Internet. For each one of these databases there is a description and a specific identification. Examples of attacks databases are: Bugtrap[04], Cve[05], Nessus[06], Arachnids[07], etc.

The Ident class has four attributes:

- name: required. The database name. Examples: Bugtrap, Cve, etc;
- ident: required. The attack identification on this database;
- pdate: required. The date on which the attack information was registered in this database;
- u_date: optional. The last updated information about the attack.
- The Ident class has one simple aggregated class that is defined here:
- url: one or more. STRING. The url that one person can use to find information inside the database.

3.2 The More_Information class

The More_information class is used to determine where the security administrator can find more information about this atack. This class can receive any information. For example, a url.

3.3 The Affected_Service class

The Affected_Service class is used to identify the service that is affected by the attack.

The Affected_Service has five attributes:

- name: required. The name of the service. Examples: Web, Ftp, Telnet, etc;
- first_group: required. The first classification. Examples: for the service Web we can use Apache, IIS, Netscape, etc. The service that does not have a specific classification receive “general” for this attribute;
- second_group: optional. The second classification, if necessary. Examples: For the service Web, first classification IIS we can

use FrontPage as a second classification;

- third_group: optional. The third classification, if necessary;
- fourth_group: optional. The fourth classification, if necessary.

3.4 The Object class

The Object class is used to identify the object that is affected by the attack.

The Object class has two attributes:

- name: required. One generic name for the object;
- object: required. The object. Examples: httpd, ftpd, fingerd;

3.5 The Version class

The Version class is used to determine for a specific object which versions are vulnerable for this attack.

The Version class has one attribute:

- version: required. The version of the object that is vulnerable for this attack. If any version is vulnerable, the value “any” appears in the attribute.

3.6 The Affected_Platform class

The Affected_Platform class is used to determine for a specific object/version which are the platforms that are vulnerable for this attack. Some attacks are only applied for some specific platform.

The Affected_Platform class has three attributes:

- name: required. The platform name. Examples: Intel, Hp, Sun, etc. If any platform is vulnerable, the value “any” appears in this attribute;
- model: optional. The platform model;
- manufacturer: optional. The manufacturer of the platform.

3.7 The OS class

The OS class is used to define for a specific object/version/platform which are the operational systems that are vulnerable for this attack. Some attacks are targeted only to some specific operational systems.

The OS class has three attributes:

- name: required. The OS name. Examples: Unix, Linux, Windows, etc. If any OS is vulnerable, the value “any” appears in this attribute;
- version: required. The OS version;
- manufacturer: optional. The manufacturer of the OS;
- release: optional. The current operational system release.

3.8 The Patch class

The Patch class is used to define for a specific object/version/platform/os which are the patches that are vulnerable for this attack.

The Patch class has one attributes:

- patch: required. The patch that is vulnerable. If every patch is vulnerable, the value “any” appears in this attribute;

3.9 The Exploited class

The Exploited class (STRING) is used to define how the vulnerability was explored to generate this attack.

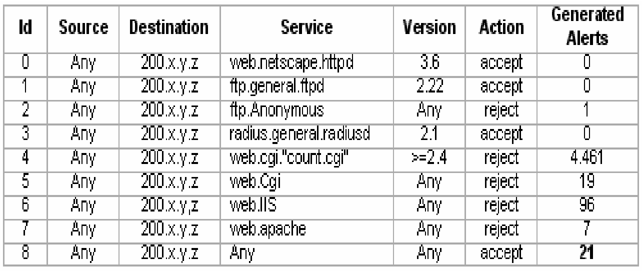
3.10 The Solution class

The Solution class (STRING) is used to define how the vulnerability can be corrected and the proceedings to stop the atack.

4 The Tests

For the tests we made two information collections in a network with approximately 50 machines using a Snort IDS sensor. This sensor was configured to generate all possible alerts.

After collecting the data we got 5 machines (10%) to create the machines caracteristcs. The profiles were converted to a group of rules. One of them is detailed below.



Id	Source	Destination	Service	Version	Action	Generated Alerts
0	Any	200.x.y.z	web.netscape.httpd	3.6	accept	0
1	Any	200.x.y.z	ftp.general.ftpd	2.22	accept	0
2	Any	200.x.y.z	ftp.Anonymous	Any	reject	1
3	Any	200.x.y.z	radius.general.radiusd	2.1	accept	0
4	Any	200.x.y.z	web.cgi."count.cgi"	>=2.4	reject	4.461
5	Any	200.x.y.z	web.Cgi	Any	reject	19
6	Any	200.x.y,z	web.IIS	Any	reject	96
7	Any	200.x.y.z	web.apache	Any	reject	7
8	Any	200.x.y.z	Any	Any	accept	**21**

Table 2 Rules based on machine characteristics

This machine has only three services running. One Netscape Web Server, one FTP Server (nom anonymous) and one Radius Server. The only cgi (common gateway interface) installed in the Web Server is the “count.cgi” (version 2.5) that is used to count pages access. The machine platform has the characteristics shown below:

Platform name: HP model: Risc

Operational System name: hp-ux version: 11.00 release: A

For all rules is applied the environment characteristics to filter the attacks that are specific for this platform/os.

Without this group of rules the security administrator has 4.605 alerts to analyze. Applying the rules remained 3 alerts with 21 occurrences.

We made the same test to five others machines. The results are shown below.

Machine	**Generated Alerts No rules**	**Generated Alerts With rules**	**% eliminations**
1	4605	21	99,54%
2	46	15	67,39%
3	38	17	55,26%
4	19	7	63,16%
5	34	7	79,41%

Table 3 Tests Resume – First Collection

Machine	**Generated Alerts No rules**	**Generated Alerts With rules**	**% eliminations**
1	2354	5	99,79%
2	218	10	95,41%
3	16	2	87,50%
4	4	0	100,00%
5	52	1	98,08%

Table 4 Tests Resume – Second Collection

5 Conclusions

This technique to reduce IDS alerts is only applied to attacks that have as target the computer/network services.

We have to do more tests in this model to find problems and improve the functionality. We think that this is one way to reduce the excessive number of alerts generated when you have to administer the security of a great network with a large number of machines.

References:

[1] Rebeca Bace and Peter Mell, *NIST Special Publication on Intrusion Detection Systems*, 2000.

[2] Object Management Group. *UML – Unified Modeling Language*, 2003. http://www.omg.org/uml.

[3] D. Curry and H. Debar. *Intrusion Detection Message Exchange Format data model and Extensible Markup Language (XML) Document Type Definition*, 2002.

[4] Bugtrap. http://www.securityfocus.com/.

[5] CVE – Common Vulnerabilities and Exposures. http://cve.mitre.org.

[6] Nessus. http://www.nessus.org/.

[7] ArachNIDS. http://www.whitehats.com.