UDC 004.056:004.77

A. FURMANOV

National Aerospace University Named After M.E. Zhukovsky, Ukraine

THE ANALYSIS OF VULNERABILITY DATABASES IN VIEW OF WEB-SERVICES ARCHITECTURE

In this article the comparative analysis of existing vulnerability databases (VDB) is carried out. The traditional architecture of Web-services is considered. One of the most popular VDB is analyzed in view of Web-services architecture.

Web-services, vulnerability database, Web service architecture, network attack tolerance

Introduction

In the present-day world it's impossible to imagine the business constructed without use of information technologies (IT). Overwhelming majority of the medium and large companies has the Internet electronic offices for active work with their customers. Usually it's the Internet-site built with use of simple and popular technologies. Nevertheless, more and more companies start to consider the Internet like a source of own market expansion for the goods and services. In the Europe and the USA electronic business have strong positions and sales volumes through Internet grow every year [1].

The analysis of network attacks for last years shows the accent was displaced to cybercrime commercializing [2]. Therefore carrying out of financial operations through the Internet dictates increased requirements to information systems safety and dependability. Each minute of business-critical systems idle time can lead to significant money and company reputation losses.

Web-services: problems of the tolerance to attacks

There are various ways to maintain the Web-services tolerance to network attacks: from the company administrative rules up to firewalls and hardware attack-detection systems [3].

However even the advanced protection means not always cope with the problem of the attack-tolerance maintenance [4]. Some architectural decisions are based on using of the multiversion approach which increase overall dependability of Web-services [5]. However they doesn't decrease vulnerability quantity in Web-services but just parry them in a case of attack. Due to that reason, it is reasonable to take into account statistics of vulnerabilities and their prevention by the developer, when a web-service component is selected.

The article purpose is to carry out the heuristic analysis of the VDB in view of the Web-service typical architecture and to choose most tolerated to attacks Web-services components.

Modern vulnerability databases

There are several popular vulnerability databases in Internet today. Most of them support common vulnerability naming standard called CVE (Common Vulnerabilities and Exposures) [6]. This fact is guarantee of vulnerability sets intersections absence in the different VDBs. Let's consider VDBs features more precisely in the table 1.

Apparently from table the most complete VDBs are Mitre (http://cve.mitre.org) and NVD (http://nvd.nist.gov). Also these VDBs are unique that gives the data in XML-format (eXtensible Markup Language) suitable for the further processing.

Despite of VDBs features similarity our choice is NVD VDB which presents the expanded information about attacks such as loss type, vulnerability type, severity of attack, range of attack, etc.

Table 1

Modern VBDs

VDB name	URL	Vulner-ability quantity	CVE support	XML -data
Mitre	http://cve.mitre.org	23,4k	+	+
Open Source Vulnerability Database (OSVDB)	http://osvdb.org	14,6k	+	
National Vulner-ability Database (NVD)	http://nvd.nist.gov	22,5k	+	+
Security Focus	http://www.security focus.com	21k	+	
BugTraq	http://seclists.org/bugtraq/	un-known		
SecurityTracker	http://www.security tracker.com	11,1k	+	
Secunia	http://secunia.com/	17k	+	

NVD VDB data structure

NVD VDB structure is determined in the XML-DTD (data type definition) document, and described in details at [6]. It consists of the following sections:

1. The general information (CVE identifier, product name, product version, product vendor, vulnerability description, publication date, detection date, severity).
2. The external links where vulnerability is described.
3. Loss type (availability, confidentiality, integrity).
4. Security protection level (user, system administrator).
5. Vulnerability type (access validation error, input validation error, design error, exceptional condition error, environmental error, configuration error, race condition error).
6. Vulnerability range (locally exploitable, remotely exploitable, initialized by the user).

Presence of all set forth above parameters in NVD VDB allows carrying out the analysis of the Web-service components separately with satisfying certain functionality criteria. For example, in case of Web-service running in the local network it makes sense to carry out the analysis in view of parameter like a «vulnerability range », etc.

For further NVD VDB analysis we developed the auxiliary program for converting XML vulnerabilities data to MySQL relational database with corresponding physical structure.

Web-services architecture

The traditional Web-service architecture consists of the following levels (fig. 1):

1. The Web-server – receives user requests and gives out to them processing results through http.
2. The Application-server - performs business-logic tasks.
3. RDBMS-server - is destined for storage and data processing.

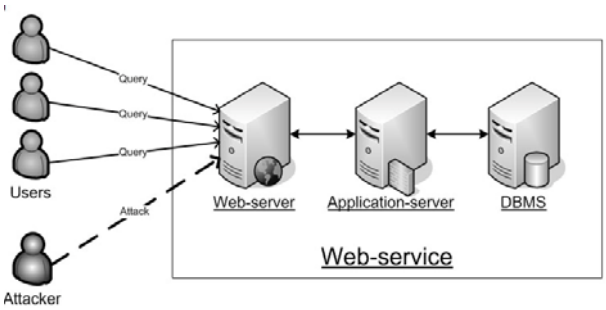

Fig. 1. Typical Web-service architecture

Let's choose some popular software as components of this architecture for each level (table 2).

Table 2

Popular software – components of Web-services

Level	Software
Web-servers	Apache httpd, Internet Information Server (IIS), Sun Java Web Server, Jigsaw, ACME thttpd, nginx
Application-servers	Apache Tomcat, BEA WebLogic, Caucho Resin, Oracle AS, JBoss, IBM WebSphere, SAP AG Web AS
RDBMS-servers	MS SQL Server, MySQL, Oracle Database, Sybase ASA, Firebird, PostgreSQL

Not always components could be integrated into single Web-service. There could have place technological incompatibility problems of the separate software. The compatibility analysis of Web-services components is considered in details in [5, 7].

VDB analysis

NVD VDB contains vulnerability on set of products from various manufacturers, since 1988. Their total at the end of 2006 made about 22.5 thousands. Processing

of large amounts of data stored in RDBMS is easier to perform using SQL-queries (Structured Query Language). Using grouping operators in combination with filters on logical conditions allows grouping the results by time intervals like years, months etc and also products, vendors. Using data from resulting datasets we shall build diagrams and analyze them.

As results of the analysis it is meaningful to choose such characteristics as:

1. Changing of VDB vulnerabilities quantity with time.
2. Distribution of products quantity by vulnerabilities quantity.
3. Changing VDB vulnerabilities quantity by the products – Web-services components.

Changing of VDB vulnerabilities quantity

Let's composing the SQL-query which selects VDB records by the month basis:

```
SELECT EXTRACT (YEAR FROM published) as y,
EXTRACT (MONTH FROM published) as m, count
(1) as cnt from t_entry group by y, m
```

As we can see from the diagram built on the received data (fig. 2) – vulnerabilities quantity in NVD VDB constantly grows. This is evidence of the constant growing of software vulnerabilities found and also VDB popularity growth.

Distribution of products quantity by vulnerabilities quantity

At the end of 2006 in NVD VDB contains the information on 8136 products from 5508 manufacturers. The SQL-query for getting out of the distribution of products quantity by vulnerabilities quantity looks like follows:

```
SELECT cnt, count (1) FROM (SELECT count
(1) AS cnt FROM t_entry GROUP BY prod-
uct_id) AS v GROUP BY cnt
```

On the assumption of received distribution (fig. 3) it's possible to make a conclusion that the VDB contains a plenty of products with insignificant (less than 10) vulnerabilities quantity. It's caused by absence of vulnerability tracking in these products owing to their unpopularity or novelty on the market.

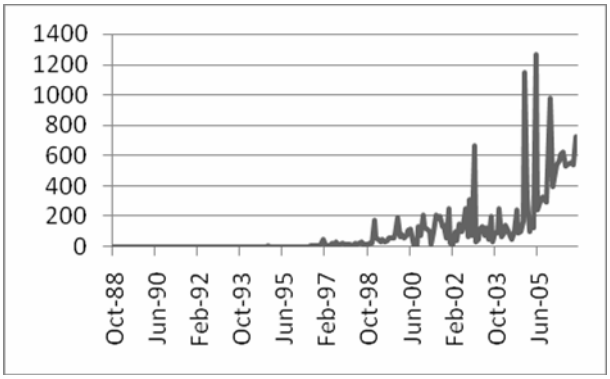

Fig. 2. Changing of VDB vulnerabilities quantity

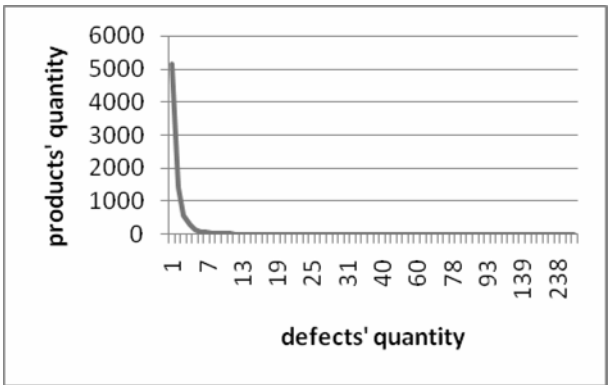

Fig. 3. Distribution of products quantity by vulnerabilities quantity

The data on such products are uninviting for the analysis and represent some type of "noise" in VDB.

Changing VDB vulnerabilities quantity by the products - Web-services components

Let's choose some components which represent the traditional Web-services architecture levels from the table 2 (Apache httpd, IIS, Apache Tomcat, BEA WebLogic, Caucho Resin, MySQL, Oracle Database, MS SQL Server) and analyze them. We shall try to discover law in change of vulnerability quantity in products in due course for what we shall take advantage of SQL-query:

```
select EXTRACT (YEAR FROM published) as y,
EXTRACT (MONTH FROM published) as m, count
(1) as cnt from t_entry where product_id
IN (select id from t_product where name
like 'product_name') group by y, m
```

Basing on the received data we shall build a series of diagrams (fig. 4 – fig. 11). We will use popular software product MS Excel for construction of diagrams. For visual display of a trend line with 6 level option of polynomial approximation was used.

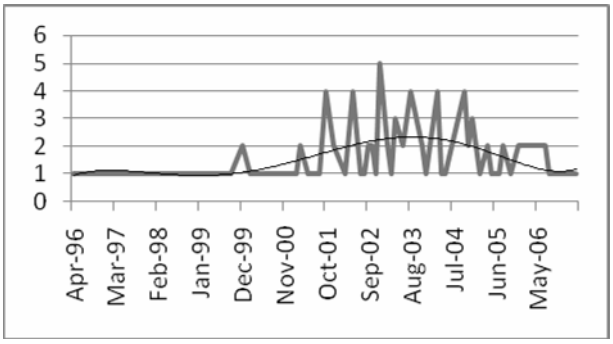

Fig. 4. Changing vulnerability quantity in Apache httpd

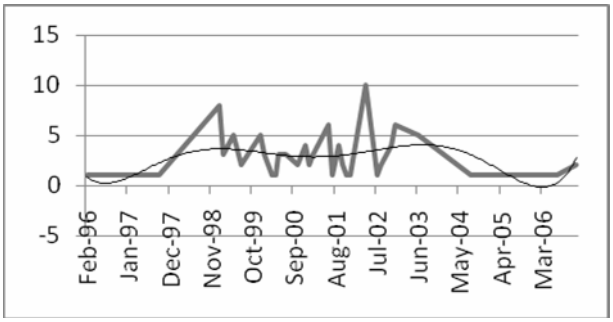

Fig. 5. Changing vulnerability quantity IIS

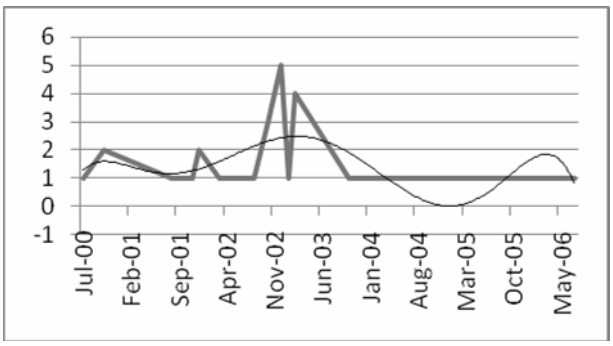

Fig. 6. Changing vulnerability quantity Apache Tomcat

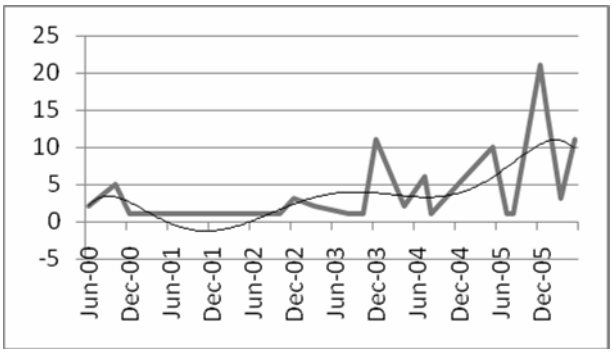

Fig. 7. Changing vulnerability quantity BEA WebLogic

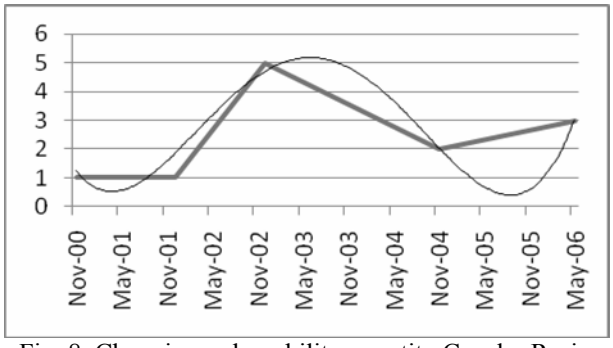

Fig. 8. Changing vulnerability quantity Caucho Resin

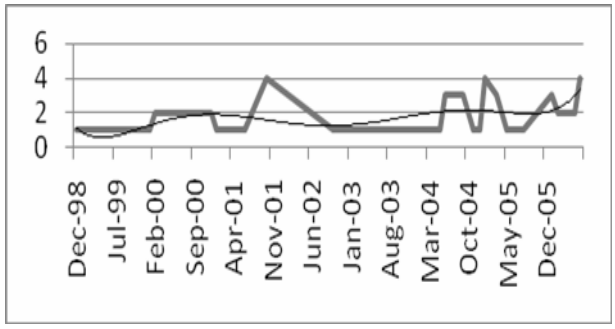

Fig. 9. Changing vulnerability quantity MySQL

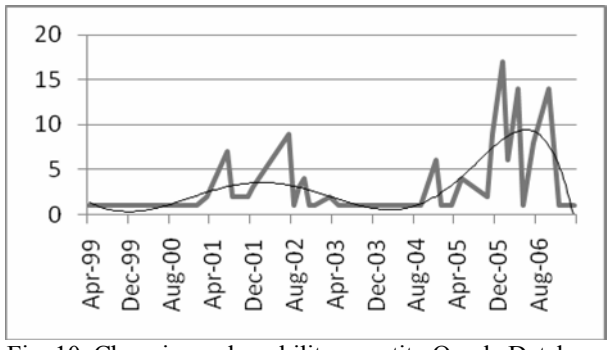

Fig. 10. Changing vulnerability quantity Oracle Database

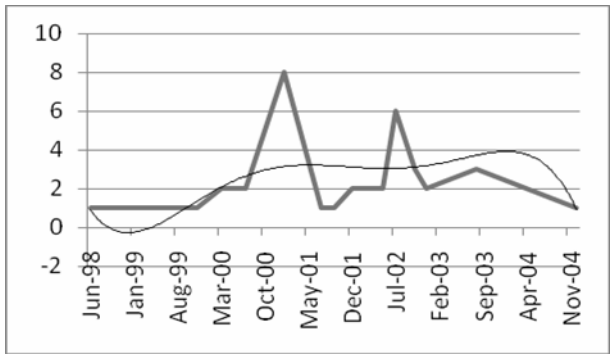

Fig. 11. Changing vulnerability quantity MS SQL Server

While analyzing the diagrams, it is necessary to note the general tendencies of change of vulnerability quantity in software. So, at the first product release moment some calm connected to a small amount of users is observed. While product popularity is growing the vulnerability quantity smoothly falls down. All subsequent peaks are caused by the new versions releases and recurrence of a detection cycle for new vulnerabilities. As shows trends of the last months some products like Apache httpd, IIS, Apache Tomcat, Caucho Resin and MS SQL Server are in a stable phase and vulnerabilities occurrences in them meets rare.

Parameters of the general and mid-annual vulnerabilities quantity for the separate products are resulted in table3.

Table 3

General vulnerability parameters by products

Product type	Product name	First release	First vulnerability	Vulnerabilities' quantity	Vulnerabilities per year (av.)
Web-server	Apache httpd	1996	1996	87	8,7
	IIS	1996	1997	102	9,3
Application-server	Apache Tomcat	1998	2000	28	4,7
	BEA WebLogic	1996	2000	104	17,3
	Caucho Resin	1999	2000	14	2,3
DBMS	MySQL	1995	1999	48	6,9
	Oracle Database	1979	1999	68	9,7
	MS SQL Server	1993	1998	48	6

Least frequently vulnerabilities are found in products such as Apache httpd (Web-server), Caucho Resin (Application-server), MS SQL Server (DBMS).

Conclusions

The analysis carried out has shows that using of VDB data it's possible to provide a choice of a Web-services component in scope of view of attacks tolerance parameters.

In the future it's planned to carry out the statistical data search on a parameter of vulnerability fixing duration in separate kinds of software and determining dependability parameter and also to carry out research of accordance given to the mathematical laws resulted in [8, 9], describing vulnerability detection and fixing.

References

1. Quarterly E-Commerce Sales [Електрон. ресурс]. – Режим доступу: http://www.census.gov/mrts/www/ecomm.html.

2. Kaspersky Security Bulletin 2006. Развитие вредоносных программ [Электрон. ресурс]. – Режим доступа: http://www.viruslist.com/ru/analysis?pubid==204007524.

3. Лысенко И.В. Основы безопасности и защиты информации в компьютерных системах. – Х.: Нац. аэрокосм. ун-т «ХАИ», 2003. – 78 с.

4. Кадер М. Типы сетевых атак, их описание и средства борьбы [Электрон. ресурс]. – Режим доступа: http://www.cnews.ru/reviews/free/oldcom/security/cisco_attacks.shtml.

5. Furmanov A., Kharchenko V.S., Gorbenko A. Intrusion tolerance of Web-systems: IMEA-analysis and multiversion architecture // Радіоелектронні і комп'ютерні системи. – 2006. – №7 (19). – С. 23-27.

6. Common Vulnerabilities and Exposures – [Електрон. ресурс]. – Режим доступу: http://cve.mitre.org/.

7. Gorbenko A., Kharchenko V.S., Tarasyuk O., Furmanov A. F(I)MEA-Technique of Web Services Analysis and Dependability Ensuring. – RODIN Book 2006. – P.153-167.

8. Sung-Wahn Woo, Omar H.Alhazmi, Yashwant K.Malaiya Assessing Vulnerabilities in Apache and IIS HTTP Servers // Proc. on 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing (DASC'06).

9. Поночовный Ю. Определение параметров закона распределения времени между отказами восстанавливаемых обслуживаемых многопользовательских систем с учётом дефектов взаимодействия // Системи обробки інформації. – Х.: ХВУ, 2004. – № 10 (38). – С.166-174.

Надійшла до редакції 12.03.2007

Рецензент: д-р техн. наук, проф. В.С. Харченко, Національний аерокосмічний університет ім. М.Є. Жуковського «ХАІ», Харків.