Web 2.0 services (vulnerability, threats and protection measures)

Jasmin Ćosić
PhD Student
Faculty of Organization and Informatics
University of Zagreb
Pavlinska 2, 42000 Varaždin, Croatia
jcosic@foi.hr

Abstract. *When at the WEB 2.0 conference in 2004, during the sessions „O'Reilly" and „MediaLive International", a concept of WEB 2.0 services was first introduced, it was clear that this concept would change the original functionality of internet [1], but only few understood how its development would make a job easier for „hackers", „crackers", „phishers" and other malicious users, and how many troubles it would bring to the experts who are in charge for information security, but also to programmers and IT managers. Struggle with Cross site scripting (XSS), SQL injection, Cross site request forgery (CSRF), malware, scareware and other types of attacks and weaknesses of web 2.0, has become very difficult, and the result uncertain. Every 15 seconds a new malicious webpage (page with malwares) is discovered in the world, [2], and up to 85% of those pages are regular and legitimate whose owners have no idea that their page was "hacked" and that the hackers embedded „malwares" into those pages. Studies have shown that the cause of the problems are new web technologies (less is attributed to vulnerability of web servers and browsers; mostly to web applications) also failures of programmers and designers, administrators of sites, where these applications are "hosted", and the web users [3].*

In this paper I will attempt to give a cross-sectional view of this field and point at the most vulnerable parts at the web 2.0 services in 2007 and in 2008. I will also try to give certain recommendations and guidelines for what needs to be paid special attention to during the IS development phases; and also how one can protect the final users, what the managers can do, what designers of application can do, as well as the administrators of the system on which the servers spin to make hackers' job harder.

Keywords. Web 2.0, Security, Vulnerability, Web application, Reliability, Hackers

1 Introduction

The power of Web 2.0 service is its interactivity, a possibility that users actively participate in the sessions, "upload" files, buy on the Internet and attend online courses; all of which is ideal for abuse.
Due to its dynamic Web 2.0 is much more than a "traditional" web which became popular with the hacking population.
For example, malicious users (read hackers) can add malware (malicious software) to the web 2.0 sites. They can also take advantage of Web 2.0 to launch "worms" which will further expand and perform vulnerable operations on a user's computer. A possibility to "upload", which is present in most Web 2.0 services, can be done in a way that a dangerous link is set to some services (wiki, facebook, etc.) which will redirect the user to pick up the malware (e.g. Trojan horses) when an uneducated user clicks on it. It will further enable him/her not only to take control over the victim's computer, but also over other computers in the network, instead of redirecting it to the desired site.
It is common to install a "keylogger" which captures users' activities on the keyboard. "Malware" can also be a code that will create "zombie" of a client's computer, which hackers can use to initiate "Distribute Denial of Services" (DDoS) or other attacks on the third user, or simply to send spam messages from the *zombie* computer. According to some sources [2], more than 70% of malicious codes today is being distributed through the RIA (R*ich Internet Applications)* and services that have been based on these technologies. The question is why this

is so, and what do designers of applications, administrators of the system on which the servers spin, and users of the system do wrong? What can be done to reduce vulnerability to a minimum and to increase security to the percentage which will be safe?

2 WEB 2.0

Web 2.0 appeared in 2004, as a logical continuation of web (1.0), and it was promoted by the O'Reilly Media Group and Media Live International [1].

The exact definition does not exist, nor it is needed. Web 2.0 represents the evolution in comparison to the "old" way of computing on the Internet. It introduced interactivity of users with services, exchange and active communication of the Web content. Unlike standard PC platforms, this platform is the Internet itself. Instead of local, computer users run Web applications on the Internet by a web browser and its content is being adapted and shaped according to the needs of users themselves.

There are many examples of its use - from business applications, e-government, entertainment, to e-education, where the web 2.0 services are the most common.

Almost all prominent universities and institutions of higher education in the world have greatly developed their e-platforms (platforms for e-education or distance learning). Behind these platforms there are usually hidden Web 2.0 services such as "moodle", "weblog", "wikipedia", various "forums, "chats", or even "youtube" and "podcasting" services, and some of the key features and technologies include AJAX, RSS, mashups, site maps, etc.

2.1 Rich Internet Applications

Rich Internet Applications (RIA) are web-based applications that were created and designed to have all the functions of "desktop" applications, but with one difference - the process of executing is divided into user's part that is being executed on the client side, and into manipulation with data that is being executed on the side of the application server[1] [4].

Rich Internet Applications are run inside the web browser, and do not require any additional software installation. The only restriction here can be the browser that is used, as well as plug-ins integrated in the browser. However, the "newer" versions of Opera, Firefox and IE and Mozilla browser work perfectly.

For security reasons, most RIA run their clients' part in a specially isolated part, the so-called "sandbox". In computer jargon, "sandbox" is a security mechanism which helps to launch applications safely. It is usually used for execution of untested codes or applications. "Sandbox" is a specific example of virtualization, because it is designed to limit clients' access to files and OS on the computer.

2.2 Asynchronous JavaScript and XML

The next thing that web 2.0 has brought is "*A*synchronous JavaScript and XML" *(*AJAX). Ajax is not a programming language such is often believed. It is technology, i.e. usage of the existing technologies in process of application development.

Unlike traditional web applications which send requests to the server and receive replies in HTML, Ajax uses web applications to receive data from the server asynchronously (intermittent) in the background. What does it mean? It means that Ajax, between HTML and the server, has "JavaScript" which calls the server, acquires data, changes and manipulates them, without obliging users to click F5-reload or refresh button.

Ajax made all web applications smaller, faster and more user friendly, but also more vulnerable!

How does it all work? In the traditional JavaScript coding we have to use GET or POST methods if we want to receive information from the database (from the server), or if we want to send some data to the server. Users always have to press "submit" to send or receive data from the server. Whenever a user sends an input data, the server returns a new page. This method is very slow and less user friendly.

Ajax makes sure that "JavaScript" communicates directly with the server through the "JavaScript XMLHttpRequest"object.

With this request, the page calls the server and receives a response from it, without doing constant "reload" of the page.

The user is on the same page and is not able to search the response from the server or to send the data to the server, because the whole process takes place in the background. A good example of Ajax application that we have to mention is "Google maps". The only thing a user needs to do is to move the mouse across the map and the map is automatically loaded, moved, increased, etc. Does it sound too good to be true?

3 Vulnerabilities of web 2.0 service

There are several main problems in development of Web 2.0 applications. The most important assumption that most developers have to keep in mind is to "never believe a client and what he or she submits in the application" [5]. There is a simple reason for it. After the input, developers have no control over the data that come back from the client's browser and they do not know what happen with the code that is sent to the client. JavaScript that is sent to the client to perform an operation can very easily be changed by hackers and by doing so he or she can change the data coming to the server [6]!

[1] Overview-Rich Internet Application ,www.theopensourcery.com/xmlria.html (viewed 15.12.2008.)

The second problem is "mashups". It is the model that "mixes" and combines data and services from several different sites and displays them on the user's browser as a new service [7].
Some influential web portal owners (including Google) allow their API to be used through gateway on other web sites and that is how “mashups” function. A typical example is the usage of "Google Maps" combined with the web portals of some hotels, motels, airports or railways. Most of those dynamic web sites use gateway in order to include Google maps into their web site to help passengers to find their destination or travel route more easily.
This kind of functionality is very vulnerable to XSS (cross site scripting) attacks.
According to reports from security experts in 2006, 2007 and 2008, vulnerability of web applications was increased. Compared to 2006 when vulnerability was 65%, in 2007 was increased to 72%, while in 2008 it was 73% out of all reported and discovered flaws and vulnerability that belonged to Web technologies. PHP is responsible for 40% of all written applications. XSS and SQL injection record a steady growth, and they were seen as the major threat to the web in 2007 and 2008 [8].

3.1 The most frequent vulnerability

During 2006, 2007 and 2008 the most frequent attacks and vulnerability methods of Web 2.0 were: SQL injection, XSS and CSRF and with a tendency to grow.

3.1.1 SQL injection

SQL injection is a great potential danger. This kind of attack uses SQL sequence from the Structured Query Language (SQL) which is a structural language for inquiries in the database. The Web 2.0 services use SQL statements in order to authenticate users by the application, to check the role and rights to access the database and to link with other database objects. [9] [10]
Web applications use the data sent from the client’s side to place an inquiry to the database server. If the data prior to creating SQL inquiry is not properly processed, some malicious samples or even systematic commands can be inserted which arise while executing the arbitrary SQL[2] [11] [12] [13].
A simple SQL statement can give data from the database to an attacker. There is a simple reason for it. The application is made in a way that it does not check (validate) input through the input field prior to processing. WHERE clause, which normally serves to set out a condition in order to limit *output* from the database. For example, if we use WHERE *Iduser* = 123 with OR'1 '='1', it will include much more than that. What does that mean?

*SELECT * from user WHERE (strUserName = 'Jasmin' OR'1 '='1') AND (Password = 'something' OR'1' =' 1');*

Statement *'1 '='1'* always gives us "TRUE" value and WHERE clause has no effect which means that SQL inquiry is an equivalent to the following:

*SELECT * from User*

SQL injection attack is performed from the address bar on the browser, from the fields to enter in forms, or from inquiries (search) and search on the sites. The biggest abuse is when hackers log into the service, without knowing the valid username and password. 2007 and 2008 were remembered by vulnerability of Web 2.0 service on XSS with 23% and SQL injection with even 34% out of the total percentage. [2] [3]

3.1.2 Cross site scripting (XSS)

Like SQL injection, XSS is associated with the unwanted data flow[3]. [14]
With XSS attack, an attacker inserts his or her malicious code into the existing and dynamically generated web pages[15].
When a user, a potential victim opens the page on his or her browser, malware is executed on the computer. The code is usually used to take control over the computer, to steal data or to change settings and firewall.
The easiest way to insert malware which will allow XSS attack is through various "guestbook", "login" forms or "forums" which allow users to include an expanded HTML or JavaScript.

3.1.3 XSRF / CSRF

Cross site request forgery - CSRF (or XSRF) is a type of attack where an attacker uses vulnerability of the web sites that believe its users or clients[15]. CSRF attacks "betray the trust that a website has in its users" [16]. CSRF attacks are far more dangerous, unpopular and much harder to defend from XSS attacks.

CSRF attacks are characterized by the following:
a) Betray the trust that site has for a certain user (registered users)
b) Include Web sites that rely on users identity (different e-banking, e-government or e-education applications)
c) Execute HTTP requests selected by attacker ("convincing users to send HTTP requests on behalf of attacker).

[2] SPI Dynamics, “Web application security assessment”, Whitepaper, 2002

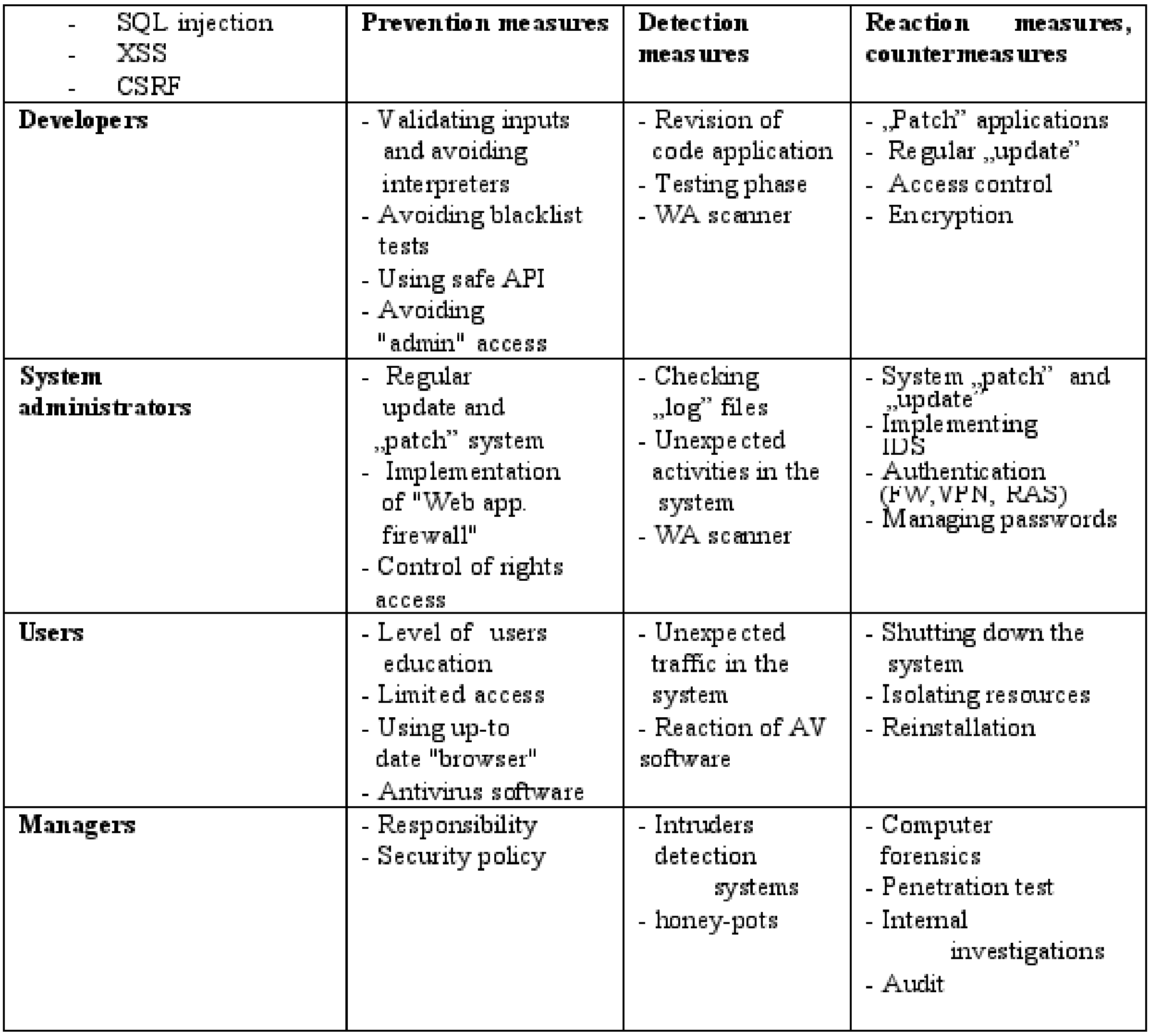
[3] CGI Security “The Cross Site Scripting FAQ” , available http://www.cgisecurity.com/xss-faq.html (viewed 15.02.2009.)

If the user has an open session with that particular web site (e.g. e-banking application) and then opens a new session in another window or visits another site, the attacker uses the second session to send a command to the first site, and pretends to come from the "real" user who is trusted.

A PC user who is trusted is used as a means of attack to the site that trusts the user. Those are usually sites with *e-commerce*, *bank* and similar applications. Attackers use the IP address of the computer that is trusted by applications, by using cookies which are located on the user's computer.
That is ideal for passing through the firewall. Firewall can barely detect such attacks and attackers use them to steal money from accounts, steal data, purchase with victims' credit cards and ultimately to change the firewall parameters in order to enter the defended network more easily next time. This type of attack is not mentioned in many safety reports, but it is necessary to pay attention to it because many application developers dedicate their time to XSS neglecting this danger. [10] [17]
2007 and 2008 were remembered by vulnerability of Web 2.0 service on XSS with 23% and SQL injection with even 34% out of the total percentage. [2] [3]

3.1.4 Malware & Scareware

The great danger that expands is the way of disturbing *"malware"*. It is the term that describes a malicious code that damages computers in every possible way. The number of attacks on systems and malware is increasingly growing and today every 15 seconds a new malicious web site is discovered in the world [2]. Five new "scareware" web sites are identified every day and the U.S.A. got ahead of China and today it is one of the top countries that host malware (37%). It is followed by China with Hong Kong (27.7%) and Russia (9.1%) [2].



- SQL injection - XSS - CSRF	**Prevention measures**	**Detection measures**	**Reaction measures, countermeasures**
Developers	- Validating inputs and avoiding interpreters - Avoiding blacklist tests - Using safe API - Avoiding "admin" access	- Revision of code application - Testing phase - WA scanner	- „Patch" applications - Regular „update" - Access control - Encryption
System administrators	- Regular update and „patch" system - Implementation of "Web app. firewall" - Control of rights access	- Checking „log" files - Unexpected activities in the system - WA scanner	- System „patch" and „update" - Implementing IDS - Authentication (FW, VPN, RAS) - Managing passwords
Users	- Level of users education - Limited access - Using up-to date "browser" - Antivirus software	- Unexpected traffic in the system - Reaction of AV software	- Shutting down the system - Isolating resources - Reinstallation
Managers	- Responsibility - Security policy	- Intruders detection systems - honey-pots	- Computer forensics - Penetration test - Internal investigations - Audit

Table 1: Security measures recommendations

4 Methods of protection

Preventing XSS, CSRF and SQL injection attacks, fighting against malware and other dangers means and requires a consistent approach at all application development stages, as well as in the entire information system. Web developers have the key role here and they can reduce the measure to the minimum from the very beginning, then web system administrators (CMS, web portal, forums, etc.), as well as users of web service who have to be educated, but also IT managers whose role must not be neglected. The prevention itself must be implemented to both the server (web developers and administrators) and the client (users) [16]. Security IS (S_{IS}) can be presented as:

$$S_{IS} = S_{PE} \cup S_{PH} \cup S_{LO}$$

Where S_{PE} - Personnel Security
S_{PH} - Physical Security
S_{LO} - Logical Security

Considering the fact that 73% of recent "intrusions" into the systems have been made by failures that happen in web technology and web applications [8], this article emphasizes S_{PS} (personnel security).
Table 1. recommends certain measures regarding vulnerabilities of Web 2.0 applications:

4.1 Programmers - developers of Web 2.0 applications

Prevention itself includes checking the entrance - it is necessary to use standard checking mechanisms of all input data and careful output coding - it is necessary to ensure that all information given to HTML by users are coded, and to avoid *blacklist* checks (by checking the list of invalid entries) in order to detect XSS at the entrances or when encoding the outputs. In order to protect the application from preventing insertion of any kind of information in the database, it is necessary to avoid interpreters whenever it is possible. If it is necessary to use the interpreter, the basic method to avoid vulnerability is to use a safe API. It is also necessary to pay attention and reduce access to databases or other background systems and to be careful when using SQL stored procedures, because they may be modified by attackers and not used for dynamic queries.
During the testing phase, it is necessary to do the code "review" and pay attention to the parts of the code - where is "input" in the application, as well as the output through HTML. There are also various "hacking" tools that simplify the task of finding the holes for web developers.

4.2 System Administrators

During the exploitation phase, web site or web portal administrators where the Web 2.0 services are present, it is necessary to do the regular "update" and "patch" with new versions. As the above-mentioned attacks are not detected by the "standard firewall" it is necessary to implement the "Web application firewall"[4] which will greatly detect XSS or SQL injection attacks.
System administrators are not allowed to grant authorities to network users which would allow them to run programs or commands independent of shells on the server, and web server should be used as a super user so that it can listen to the requirements from the standard 80 port. When it starts working, it will change its UID to a username that it specified in a configuration file. [18]

4.3 Users

Final users themselves should be fully educated and aware of these dangers, and they should make regular dissemination of their knowledge. Only one "wrong" click of an uneducated user who receives an e-mail with a link to malicious websites, may be enough to jeopardize security of the whole information system. Also, a very simple step that any user can take is to "never browse the Internet while using the administrator account" and to "turn off scripting support in his or her browser "[19]

4.4 Managers

The role of IS manager is to adopt security policies and principles, as well as internal documents necessary for the proper functioning of an IS. Each IS must have its own developed model which has to predict any possible situation. The manager’s responsibility is crucial when selecting IT staff. The manager has to know when and whether it is necessary to do IT forensics, penetration testing or internal investigations. A security plan review is necessary to do in certain time intervals.

5 Conclusion and recommendations

Cyber space became more useful by developing Web 2.0. This has also brought dangers that came up to the surface. Users who were comfortable in the former environment became victims without even being aware of it. It was easier for malicious users and experts in charge of information to deal with XSS, SQL injection, CSRF, malware, scareware and all other types of attacks and weaknesses of Web 2.0. on

[4] The Open Web Application Security Project , Web Application Firewall, available http://www.owasp.org/index.php/Web_Application_Firewall (viewed 12.01.2009)

the daily bases. Every 15 seconds a new malicious webpage (page with malwares) is discovered in the world, and up to 85% of those pages are regular and legitimate. This article's goal was to encourage developers, IT experts and managers in particular to think in order to address information security that is put in the background, and to become aware of the greatest dangers that threat to any information system. The article also tried to give specific instructions to developers, administrators and final users in order to increase the number of steps that a malicious user has to pass to reach our information system. The author also pointed out the things that we need to pay attention to and gave some recommendations that could be used and installed into the individual model of development and success of a "safer" IS. Due to the rapid Internet development in the last several years, companies should definitely do the "update" of their documents called "Security Policy" and "Principles of IS Security". It is also recommended to do the detailed audit of all information systems - "penetration testing", checking whether their system is vulnerable to these attacks and to what extent. The most important thing in the whole process is implementation of IS security measures and policies. Only one exception or failure can be fatal to the whole system.

References

[1] O Reilly: What is WEB 2.0, Design Patterns and Business Models for the Next Generation of Software, available `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html` (viewed 12.12.2008)

[2] SOPHOS: Security threat report 2009, available `http://www.sophos.com` (viewed 12.01.2009)

[3] CENZIC, „Click to secure", available `http://www.clicktosecure.com/news_events/Cenzic_AppSecTrends_Q1-08.php` (viewed 15.12.2008)

[4] T.Jithka, "Rich Internet Application (RIA)", Journal of Computing Sciences in Colleges, Papers of the Nineteenth Annual CCSC South Central Conference, 2008

[5] CGI SECURITY, The Cross-Site Scripting (XSS) FAQ, available `http://www.cgisecurity.com/xss-faq.html` (viewed 19.12.2008)

[6] ACUNETIX, Cross Site Scripting - XSS - The Underestimated Exploit, available `http://www.acunetix.com/websitesecurity/xss.htm` (viewed 19.12.2009)

[7] What is a Mashup, available `http://www.tech-faq.com/mashup.shtml` (viewed 15.01.2009)

[8] CENZIC, Quarterly Trends Report, available `http://www.cenzic.com/index.php?id=resources_reg-not-required_trends` (viewed 15.01.2009)

[9] Zeller W., Felton E. CSRF: Exploitation and Prevention, Princeton University, 2008 , Available : `http://www.freedom-to-tinker.com/sites/default/files/csrf.pdf` (viewed 15.01.2009)

[10] Ethical Hacking and Countermeasures, EC-council, 2004

[11] Anley C. „Advanced SQL injection in SQL Server Application". An NGS Software Insight Security Research (NIRS) Publication , 2002

[12] Cesar C. "Manipulating Microsoft SQL Server using SQL injection". Whitepaper, 2002

[13] Yao-Wen Huang, Shih-Kun Huang, and Tsung-Po Lin ,Web application security assessment by fault injection and behavior monitoring, Proceedings of the 12th international conference on World Wide Web, Budapest, Hungary , 2003

[14] Apache, „Cross Site Scripting Info", `http://httpd.apache.org/info/css-security`

[15] G.Wasserman, Z.Su, "Static Detection of Cross site scripting vulnerabilities", Proceedings of the 30th international conference on Software engineering, Leipzig-Germany, 2008

[16] Zeller W., Felton E. CSRF: Exploitation and Prevention, Princeton University, 2008 , available `http://www.freedom-to-tinker.com/sites/default/files/csrf.pdf` (viewed 15.01.2009)

[17] A.Barth, C.Jackson, J.C.Mitell, „Robust defenses for CSRF", Proceedings of the 15th ACM conference on Computer and communications security, CCS 2008

[18] Bača M., „Uvod u računalnu sigurnost", Narodne novine , Zagreb, 2004

[19] Adam N. & M. Damodaran, „Security in Web 2.0 application development",iiWAS '08: Proceedings of the

10th International Conference on Information Integration and Web-based Applications & Services,
Publisher: ACM,2008

[20] G.Wasserman , Z.Su, „Static Detection of Cross-Site Scripting Vulneraiblities",
ICSE '08: Proceedings of the 30th international conference on Software engineering ,
Publisher: ACM,
2008