An Ontology of Security Threats to Web Applications

Dr. Ernest Cachia, Mr. Mark Micallef
Software Engineering Process Improvement Research Group
Department of Computer Science and Artificial Intelligence
University of Malta
ernest.cachia@um.edu.mt, mmica01@um.edu.mt

Abstract

As the use of the internet for commercial purposes continues to grow, so do the number of security threats which attempt to disrupt online systems[1][8][9]. A number of these threats are in fact unintended[11]. For example, a careless employee might drop a cup of coffee onto essential equipment. However, when compared to the brick and mortar world, the internet offers would-be attackers a more anonymous environment in which to operate. Also, the free availability of hacking tools makes it possible even for the curious teenager to carry out dangerous attacks[3]. Despite this ever-present threat however, it is all too often the case that security is dealt with (if at all) after a web application has been developed[2]. This is mainly due to our software development heritage whereby companies prefer to focus on the functionality of new systems because that provides an immediate return on investment.

As a precursor to proposing an framework for building security into web applications, this paper presents an ontology of threat to web applications. The thinking behind this is that much the same as in the military world, one needs to have as much intelligence about the enemy as possible, the same can be argued in the case of online security threats. Such an ontology would enable stake holder in online applications to take less of a reactive stance but instead be more proactive by being aware what's out there.

Keywords: *Security, Software Quality Assurance, Web Applications, E-Commerce*

1. Introduction

W. Edwards Deming stated several years ago that "the quality of a product is directly related to the quality of the process used to create it"[4]. Although Deming's work involved production work during World War II, his statement holds true today when dealing with complex software systems. As part of a wider body of work dealing with the rapid development of high-quality e-commerce systems[6], the authors of this paper identified the five most important quality attributes in e-commerce systems of which the most important was deemed to be security[5].

Despite the importance of security, it is still often the case that high-profile breaches surface in the news with many more going unannounced[3][8][9]. It appears that development companies focus mostly on functionality when developing a system since this is perceived to provide an immediate return on investment. However, a study amongst 350 online shoppers, by the authors of this paper revealed that 86% of potential customers would not shop at a site if they were not confident in its security capabilities[5]. Also, 36% of online shoppers consider security considerations as the primary reason for choosing a brick and mortar store over an online equivalent[5]. This leads to the observation that web application developers are not really delivering a product of good quality when they concentrate on functionality, but rather they deliver one of perceived quality. The lack of focus on security throughout a web application's development life cycle often leads to a vulnerable first release of that application[2]. Considering potential customers' security awareness, this may prove to be a costly mistake.

Through research in psychology, it has been established that humans are able to handle large quantities of information much more efficiently if they are able to classify it into a manageable number of categories[15]. In light of this, the authors of this paper argue that managing thousands of security threats would be far easier if each threat could be seen within the context of a category of similar threats. Once a threat has been placed in this context, it is far easier to manage it by applying the same treatment that would be applied to similar threat albeit slightly tailored to any particular characteristics of the individual threat.

In this paper, it is being argued that there is a need for

a security ontology of online security threats. The authors also go on to propose such an ontology.

2. Overview of the proposed ontology

At its highest level, the ontology consists of 6 components as shown in figure 1.

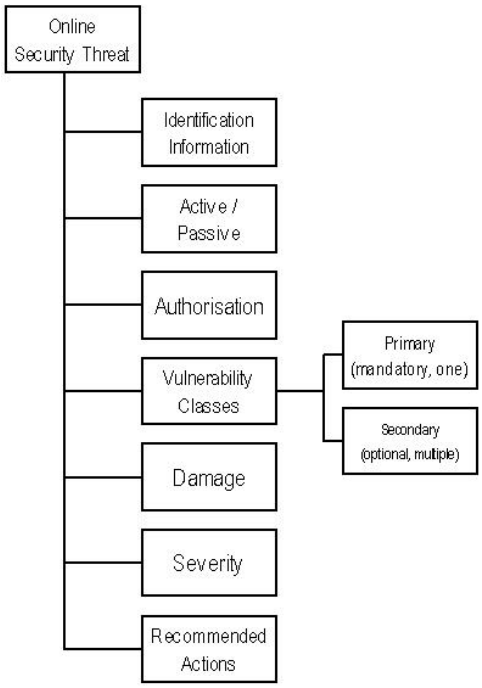

Figure 1. A high-level overview of the proposed Ontology

Each of these components is discussed in the following sections.

2.1 Identification Information

Every security threat needs to be identified. The *identification information* component of the proposed ontology provides for this. It is being proposed that threats be given a unique identifier (for use in database storage of information), a name, and a freetext description. This is depicted in figure .

2.2 Active/Passive Threat

The second component of the proposed ontology seeks to indicate whether a threat is *active* or *passive*. An active threat is in essence an intentional threat by someone with malicious intentions. This may include a hacker intentionally trying to gain access to credit card information, a virus trying to corrupt your operating environment, and so on. On the other hand, a passive threat is unintentional. That is to say it happened by accident. Instances of unintentional threats may include dropping a coffee mug over a server, maintenance personnel disabling power lines to essential equipment, and so on.

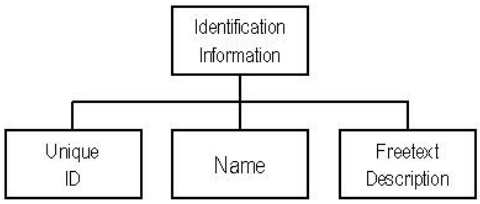

Figure 2. The *Identification Information* Component

2.3 Authorisation

The authorisation component seeks to identify what type of user would carry out a particular attack. Three possibilities exist and these are shown in figure **??**. Firstly, a user could be *authorised*. This means that a user has been delegated rights in the environment, on a system, or on a network and chooses to abuse of these rights in carrying out an attack. The second possibility is that of an *unauthorised* users gaining access to the environment through some weakness and thus causing havoc on the system. Finally, there is also the possibility of an *impersonation*. An impersonator could be a user, hacker or even a computer program which manages to gain access by taking on the identity or rights of another authorised user.

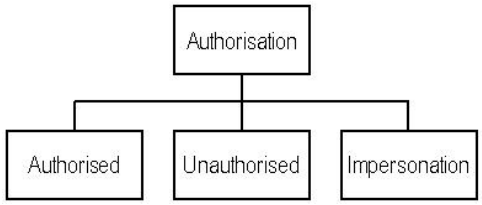

Figure 3. The *Authorisation* Component

2.4 Vulnerability Classes

The term *vulnerability class* refers to one of five classes which have been defined in this ontology as broadly covering all possible vulnerabilities which a security threat can exploit. shown in the figure 4. As a minimum, a threat will have a *primary vulnerability class* assigned to it. This refers to the vulnerability which is most exploited by the threat. Should a threat exploit more than one vulnerability, there is the option to define additional *secondary vulnerability classes*.

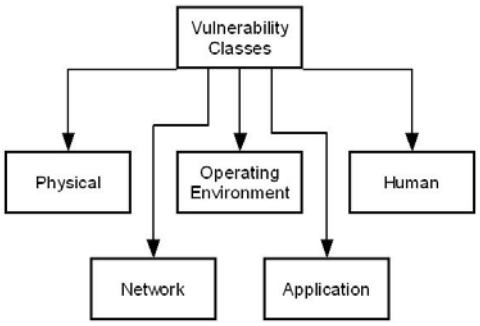

Figure 4. Types Security Threats

A web application may be compromised by threats which exploit vulnerabilities relating to any one of the categories shown in figure 4. All too often, one may be lulled into a false sense of security by emphasising security against threats on one of the above-mentioned levels. For example, one could feel that using secure communications[12] is enough to ensure a particular application's security. However, secure communications only assure privacy at the network level and a private web conversation with a hacker will not deter him/her from exploiting vulnerabilities of a different nature. The following is a brief discussion of the threat categories (sometimes referred to as levels) shown in figure 4.

At the ***physical level***, one must ensure that the locations where hardware (be it for deployment or development purposes) resides are physically secured[16]. Any physical intrusion into such a location may result in interruption of service, stolen data, and so on. In such cases, intruders may not necessarily be outsiders but could also be insiders (e.g. disgruntled employees).

Another category of security threats involves exploiting vulnerabilities at the ***network level***. Such threats attempt to manipulate shortcomings in communication protocols to disrupt service and/or gain access to private information. Appropriate actions must be taken to prevent this from happening[11].

The ***operating environment*** where a web application is hosted is also vulnerable to attacks. Vulnerabilities may be present in any components ranging from the operating system [17] to any other web-enabled component such as web server software, mail server software, and so on[18]. Again, one can easily imagine a scenario where an application is highly secured by developers, only to be breached by attacking its operating system instead of the application itself.

The category of ***application level*** threats refers to threats which exploit vulnerabilities within the web application itself. SQL injection attacks, cross-site scripting and authentication related attacks are typical examples of threats which can be found at this level[14][19]. It is beyond the scope of this paper to explore the technical merits of such attacks.

Finally, one should discuss vulnerabilities at the ***human level*** of security. Such vulnerabilities involve trusted human beings knowingly or unknowingly enabling outsiders to breach a web application and access restricted data[20]. Numerous incidents have been reported whereby a technically secure web application was breached because an insider was tricked into revealing information such as their username and password (a technique commonly known as social engineering). An effective security policy must ensure that trusted users are in a position to repel any attempts made to use them as a weak point within the application's security structure.

2.5 Damage

The *damage* component of the proposed ontology seeks to identify what sort of damage would be caused by the threat should it succeed in materialising. The following possibilities have been extracted from a US government report on information security risks[7]. Should new types of damage surface in future, these should be included in the ontology. A security threat can cause damage in one or more of the following categories:

- Disclosure of information
- Modification of information
- Destruction of information
- Degredation of service
- Denial of service
- Website defacement

2.6 Severity

The *severity* component gives an indication of the amount of harm and/or fallout which would result from the threat if it materialises. It is proposed that this information be categorised as *low*, *medium* or *high*. A *low* severity level indicates that the consequences of a materialisation of this threat would have little repercussion on the web applications ability to keep functioning and on its reputation. An example of this might involve a relatively harmless adware program managing to install itself on a server. A threat with *medium* severity is one that causes a degradation in service without completely disabling the site or one that causes damage or loss which can be recovered. Examples might include minor website defacements, or loss of data which is regularly backed up and can thus be retrieved. Finally, a *high* severity level indicates threats which severely impinge on a web application's stability and/or its users' privacy and security. Examples include hackers gaining access to clients' personal information, viruses corrupting or erasing data in such a way that it cannot be retrieved, an so on.

2.7 Recommended Actions

Finally, one should discuss the *recommended actions* component of the proposed ontology. This component has three sub-components as shown in figure 5.

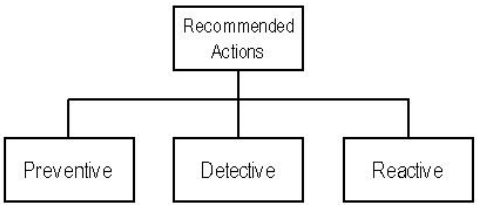

Figure 5. The *Recommended Actions* Component

The *preventive* sub-component refers to actions which can be taken to prevent the treat from materialising. A list of generic actions such as *deploy firewall solution*, *validate all application inputs*, etc is defined but due to length restrictions cannot be included in this paper. Since preventive measures may sometimes fail, the ontology also seeks to define a list of *detective* actions. These actions/mechanisms will be put in place to detect the occurrence of a security threat. Finally, the proposed ontology defines a number of possible *reactive* actions which are to be taken in the case of a materialisation of a particular security threat. Again, a number of generic actions have been defined but cannot be included here.

3. Conclusion and Future Work

It is believed that the ontology proposed here will prove to be very useful once a sufficient number of threats have been defined and place within it. Future work in this area will start off with this very task. However, the final aim of this line of work within the SEPI research group is to develop a multi-tier, multi-role security framework which will be developed. This framework will provide a process based way of securing web applications against security threats which are defined within the framework proposed in this paper.

4. References

[1] Glisson W. B., Welland, R., "Web Development Evolution: The Assimilation of Web Engineering Security", Proceedings of the 3rd Latin American Web Congress, IEEE Press, 2005.

[2] Gaur N., "Assessing the Security of Your Web Applications", Linux Journal, April 2000

[3] Morrisdale P.A., "The Six Dumbest Ideas in Computer Security", http://www.ranum.com/security/computer_security/editorials/dumb/

[4] Rakitin S. R., "Software Verification and Validation: A practitioner's Guide", Boston: Artech House, 1997

[5] Cachia E., Micallef M., "Towards Effectively Appraising Online Stores", Proceeding of the Software Engineering Knowledge Engineering (SEKE) Conference, 2004

[6] Cachia E., Micallef M., "Towards a RAD Framework for E-Commerce Systems", International Conference on Web Information Systems and Technologies, 2006

[7] United States General Accounting Office, "Information Security Risk Assessment - Practices of Leading Organisations", United States Government, 1999

[8] Deloitte, "2005 Global Security Survay", Deloitte Touche Tohmatsu, 2005

[9] Gordon L. A., Loeb M. P., et al, "2005 CSI/FBI Computer Crim Survey", Computer Security Institute, 2005

[10] Hersherb J., et al "Software quality and the Capability Maturity Model", Communications of the ACM, June 1997

[11] Mackey D., "Web Security for Network and System Administrators", Thomson Course Technology, 2003

[12] Freier A. O., Karlton P., Kocher P. C., "The SSL Protocol Version 3.0", Netscape, 1996

[13] Schneider B., "Secrets & Lies: Digital Security in a Networked World", John Wiley & Sons Inc, 2000

[14] "The Open Web Application Security Project", http://www.owasp.org/

[15] Antherton J. S. Learning and Teaching: Assimilation and Accommodation http://www.learningandteaching.info/learning/assimacc.htm ,2005

[16] Diroff T.E., "The protection of computer facilities and equipment: physical security", ACM SIGMIS Database, ACM, 1978

[17] Blakely R., "Hackers Uncover Biggest Microsoft Vulnerability", TimeOnline, http://business.timesonline.co.uk/article/0,,9075-1968021,00.html, 2006

[18] Nagel B., "Microsoft Releases Out-of-Cycle Patch for VML Flaw", http://www.redmondmag.com/news/article.asp?EditorialsID=782 2006

[19] Jovanovic N., Christopher K., Engin K., "Precise Alias Analysis for Static Detection of Web Application Vulnerabilities", Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security, 2006

[20] Orgill G. et al, “The Urgency for Effective User Privacy-education to Counter Social Engineering Attacks on Secure Computer Systems”, Proceedings of the 5th conference on Information technology education , ACM Press, 2004