Understanding & Defending Against Data Breaches, as part of a Custom Software Development Process

Frank S. Rietta
M.S. Information Security
Rietta Inc, Johns Creek, Georgia

Abstract

Security incidents that lead to data breaches have been happening a lot, from the latest Anthem Blue Cross breach, to Target, to Home Depot, to breaches including the MongoHQ incident that lead to the BufferApp compromise. Waiting until a software project is complete and bolting security on through the use of security software or network security countermeasures is not effective enough. To have a chance to build a secure system, a team requires the active support of developers and for the organization to adopt a written information security policy that influences business model decisions and the requirements gathering process.

Keywords

Privacy, database security, web application security, data breach

Introduction

Application Security is the subset of Information Security that is interested in protecting data and privacy from abuse by adversaries that have legitimate access to the application as a whole. Especially when network defenses are unable to protect the system attack. Web applications are of particular concern because one cannot protect the application by blocking access from the public without preventing legitimate users from accessing the system or using their account.

Web applications are especially vulnerable to attack, but not just by technical attacks against insecurely developed applications via SQL injection, cross-site scripting, and direct object reference attacks. Many successful of the attacks are leveled against easy targets of opportunity, with upwards of 95% of incidents involving harvesting user credentials from customer devices and then logging into web applications with them[1]. According to the same Verizon report, "organized crime became the most frequently seen threat actor for Web App Attacks, with financial gain being the most common of the primary motives for attacking." However, just because your app is not a financial one does mean that your development team can ignore the fundamentals of security application development.

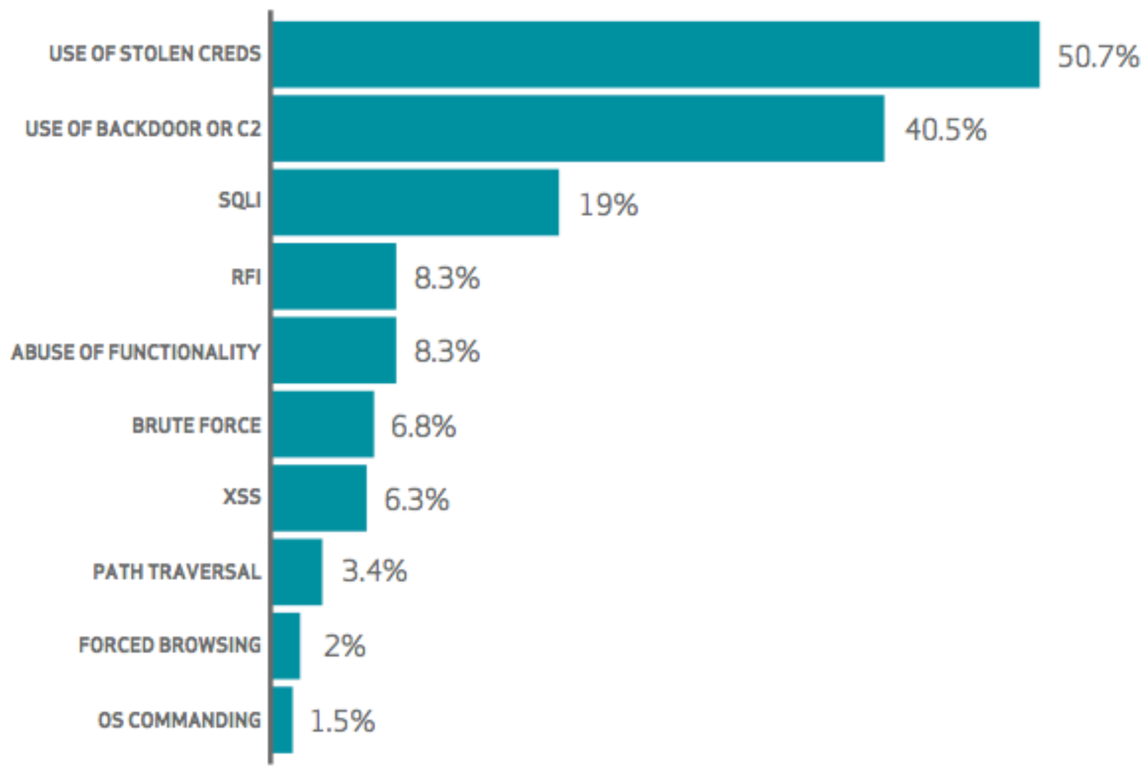

Figure 1 "Variety of hacking actions within Web App Attacks patterns (n=205)," (2015 Data Breach Investigations Report, p 41, see [1]) While use of stolen credentials is the top threat, attack through fundamental vulnerabilities, SQL injection attacks, and abuse of functionality account for a large surface area that is vulnerable across a multitude of applications. The use of backdoors to maintain access implies an Advanced Persistent Threat, wherein the attackers have already built themselves a way to maintain access.

Web application insecurity is not a new threat and it is compounded by the culture in which custom software is developed. Most applications are developed by outside teams to solve particular business or organizational requirements on tight budgetary and calendar deadlines. One of the seminal books on building security software observed that "most outsourced software (software developed off-site by contractors) is full of backdoors…. **Companies that commission this kind of software have not traditionally paid any attention to security at all**" (2004)[2]. I read that as an undergraduate computer science major eleven years ago and it has been my experience in the intervening years that this pattern has held true for every application I have ever worked with as a professional. Even for the projects for which security was a concern, the principle concern has always been functional feature development.

By definition there is a data breach is a security incident in which sensitive, protected or confidential data is copied, transmitted, viewed, stolen or used by an individual unauthorized to do so[3]. It is also required to consider an incident a breach if the unauthorized person had access to the data and the means to read it because the burden of proof is on the breached entity to prove that the data was not stolen, which is exceedingly difficult if not impossible.

A Litany of Data Breaches

The year 2014 may well be remembered as the year of the data breach with some estimates that over a billion records of personally identifiable information (PII) were compromised[4].

Visualizing the size and scope of data breaches based upon media reports is difficult and it is easy to forget about past events as the news cycle moves. However, there are efforts such as the visualization in Figure 2, which is based upon the breach data curated by http://databreaches.net.

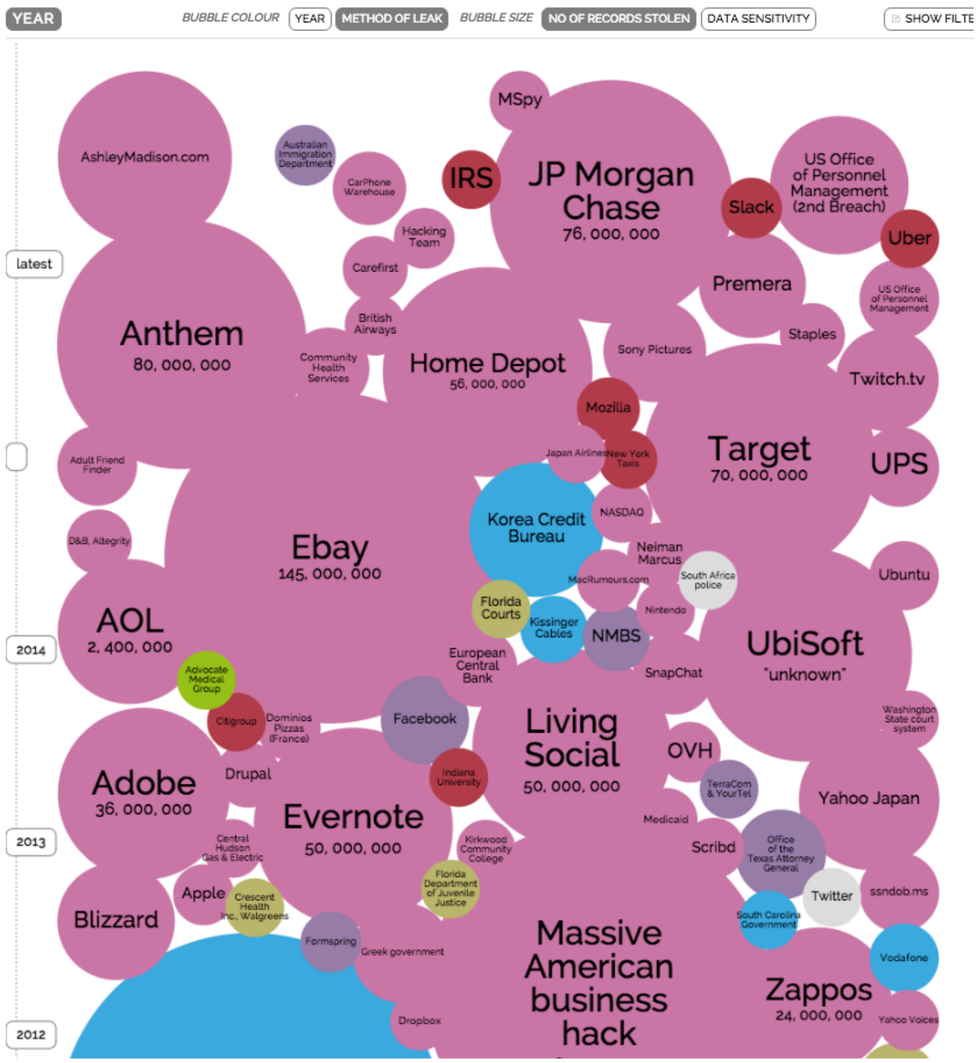

Figure 2 Portion of a visualization called the *World's Biggest Data Breaches* (August, 2015). The size of each circle represents the number of records breached, which is an approximation for the number of people impacted. Magenta circles represent a breach that is attributed to deliberate hacking, as opposed to an inside job or negligence on the part of employees.[5]

Experiencing a data breach as a company leads not just to loss of goodwill after being required to issue a mandatory announcement, but also significant legal cost, especially subsequent to the "outcome of an FTC case against Wyndham Worldwide, consumers might have a bit more support. An appeals court ruled in favor of the FTC, which accused Wyndham of 'failing to safeguard consumer data.' What it means is that under certain circumstances, the FTC can penalize companies for not implementing a certain standard of security"[6]. Regulations are likely to increase and there may soon come a time where senior executives will face criminal sanctions, including incarceration, for gross negligence in security matters, similar to Sarbanes-Oxley in corporate financial matters.

There are far too many breaches in the news to treat each of them specifically, but here are some of the incidents that are of particular note for the purposes of understanding data breaches and doing something about them. For example, breaches may result from

- Compromised staff credentials, which would be preventable by two-factor authentication
- Automated technical exploits that are aggressively applied over a large number of sites, the online equivalent of attackers trying every door to see which is unlocked
- Poor security, including unencrypted backups, that leading to an unauthorized person having access to both the data and the means to read it

What is security?

When I was a graduate student, a professor told the class that "Security is not a functional requirement" and in a sense he was right from a classical project management point of view. But the danger is that treating security as a secondary requirement all too often relegates it to irrelevance. Technical controls and business model changes to minimize sensitive information are constantly put off until such a time as the application or computer system is in deployment, sensitive data is amassed in a database unprotected, and the technical debt related to fixing the vulnerability becomes so massive as to be unsolvable given an organization's budgetary constraints.

The role of security is to protect the confidentiality, integrity, and availability of data and information resources. Confidential information is protected by reasonable technical security countermeasures, also known as technical security controls[7]. To choose appropriate security countermeasures requires a detailed threat model and to do so means a comprehensive understanding of the nature of the information handled by the system. For this purpose, it's vital to have a classification system as a fundamental planning tool.

One thing that security is not is an on/off switch. One cannot simply add SSL to a web server to encrypt data to and from the application while it is in transit. Sensitive data has to be protected in all of its information states[a], such as while it is in

- Transit over the network
- Storage in databases, files container, or in backup (called Data at Rest)
- Processing, such as while being accessed by users or staff members in the ordinary course of their jobs

Applying a variety of technical and business countermeasures often achieve a reasonable level of security as part of a defense in depth effort. Ask what is the worst thing that can happen and then plan accordingly.

[a] John McCumber introduced this model in 1991 with *The McCumber Cube*, which depicts information states and security tradeoffs as a three-dimensional Rubik's Cube-like grid. Information about this approach can be found online and also in his 2004 book *Assessing and Managing Security Risk in IT Systems: A Structured Methodology*.

Ask what is the worst that can happen

You do not have to wait for a real attack on your production systems to understand the scope of a data breach when your organization is to face a compromise. Simply point to a server on your network, or in your cloud-based control panel, and declare that it has been completely compromised by an unauthorized outsider. Then talk with your developers and business team to understand the scope of access that the malicious user now has access to in this attack scenario.

If the compromised server is a web server, presume that the database credentials used by the application are compromised and that the attacker has the ability to make arbitrary changes to the web application for his or her own malicious purposes. Then ask the following questions:

1. Do you have a data breach in this case?
2. If so, what can be done to minimize the scope of that breach as part of your systems design and implementation?
3. Can the business model be changed to not require as much legally sensitive data as it has now because data that you do not poses cannot be stolen from your system.

Systems can be designed to minimize and compartmentalize sensitive information such that the compromise of a particular server does not mean that the entire application is breached. In some cases, legally protected information can even be encrypted such that a complete database dump does not mean that there has legally been a data breach. However, this depends entirely on proper encryption key management with private decryption keys that have never been present on any of the compromised servers. However, even if such extraordinary design cannot be applied to your situation, working through this exercise with your team throughout the design and development process is extremely valuable.

Commercial information classifications

As I have previously published online, it is extremely important to understand the type of information that your application processes in order to properly understand the technical or other protection necessary to prevent a data breach[8].

You may be familiar with information classification in the military, you hear about secret or top secret information. In a commercial application, there are other categories of information. The five categories are Public, Internal Use, Confidential, Sensitive, and Highly Sensitive.

1. **Public:** Public information
2. **Internal Use:** Confidential business information

3. **Confidential:** Information that customers consider confidential

4. **Sensitive:** Personal and Private Information (PII), information that THE LAW considers confidential
5. **Highly Sensitive:** Encryption keys, server secrets, staff/admin passwords

All too often, business owners and startup founders think they can choose the level of security for their applications. But the truth is that only the first two categories, Public and Internal Use information fall within their discretion. Above the yellow line, you have some wiggle room. Between the yellow and the red line is the uncanny valley of personal information and below the red line enhanced security countermeasures are highly advised.

Most business information is Internal Use, most customer data is Confidential, unless it is legally defined as PII and then it is Sensitive. Encryption keys and staff passwords are always Highly Sensitive because of the extraordinary access typically granted through these keys and credentials (take look at Figure 1, which shows over 50% of attacks involving stolen credentials).

Data breach laws

While having a solid internal classification system of sensitive data is important, some data is more important to protect because it is defined as sensitive as a matter of law. It is simply not up to individual companies to decide to protect this information or not.

State and Federal law prescribe encryption for any system that deals with legally protected personal information, for example according to Georgia Law (OCGA 10-1-911):

> "Personal information" means an individual's first name or first initial and last name in combination with any one or more of the following data elements, **when either the name or the data elements are not encrypted or redacted**:
>
> - Social security number;
> - Driver's license number or state identification card number;
> - Account number, credit card number, or debit card number, if circumstances exist wherein such a number could be used without additional identifying information, access codes, or passwords;
> - Account passwords or personal identification numbers or other access codes[9]

Because laws defining what constitutes personal information vary from state to state, systems must protect individuals' privacy and enforce privacy measures adequately without hanging too much attention on the law of a particular state. Even when your company has thinks it only has legal nexus in its home state other states may have the means to reach out and effect your organization if enough people in their state are harmed by a breach of your users' data. The National Conference of State Legislatures provides good reference for state-by-state data breach notification and data disposal laws [10, 11].

Additionally, Federal laws include the *Computer Fraud and Abuse Ac*t (CFAA), *Health Insurance Portability and Accountability Act* (HIPAA), the *Federal Information Security Management Act of 2002* (FISMA), and others. Some laws apply only to government agencies and their contractors, but others apply to any covered entity doing business within the United States.

Treating security as a requirement

Because application security must begin before the first line of code is written, a security-based development approach considers security requirements as a fundamental part of the design process. These become considerations that are attached to the use cases or user stories, which is an approach used by many teams to shift the focus from writing about requirements to talking about them[12].

I like the user story model because it is something that the team can begin to write casually while discussing the functionality. User stories are just two ways to capture functional requirements for a software project, until you realize that they can also be used to capture a malicious users' intent.

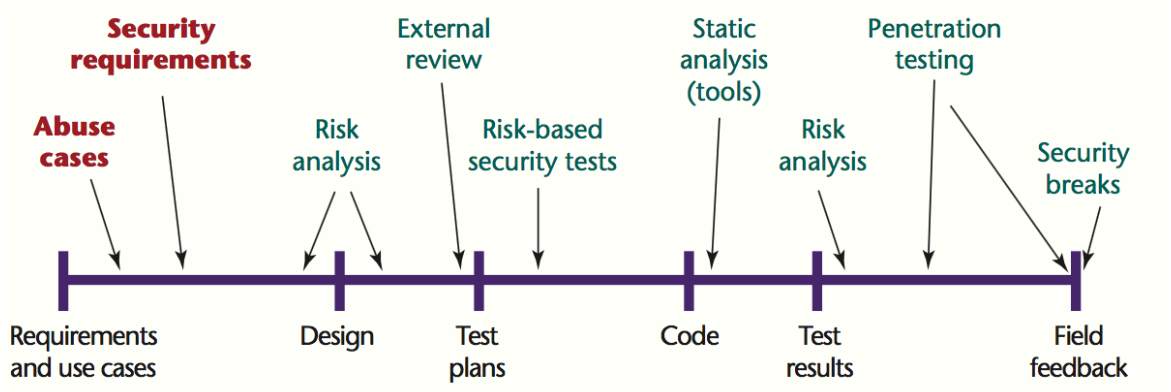

Figure 3 "The software development life cycle with abuse cases and security requirements" as a fundamental part. From page 91 of the May-June, 2004, issue of IEEE Security & Privacy, see cite 12.

Abuse stories are an adaptation of the abuse cases that were introduced by Hope, McGraw, and Anton to address the fact that there is no convenient security pull-down menu that will let you select "security" and then sit back and watch the magic happen. Instead your requirement gathering process must go beyond considering the desired features. Every time a feature is described someone should spend some time thinking about how that feature might be unintentionally misused or intentionally abused [13, 14.] One should also note special security concerns when writing happy path user stories so that the developer can be in the mindset of addresses these concerns when he or she picks up that story for implementations.

Examples of User Stories with Security Concerns

- As a Pawn Shop Clerk, I scan a copy of the customer's driver's license because the company is required by law to keep this record at least two years from the date we purchase a used valuable from a customer.
  - Security Notes:
    - Driver license numbers are PII according to State law (OCGA 10-1-911)
    - These records should be encrypted because lawyer says that gives us an out if the record is copied by an unauthorized person
    - These can be deleted after 2 years. Joe thinks that Amazon S3 can automatically delete the file if we set the expiration date when storing it
- As a Visitor, I can create a new account by filling in my e-mail address and desired password
  - Security Notes:
    - Can we verify that the user really has the email address on signup?
    - The password should be at least 12 characters long and should definitely allow for spaces and punctuation
- As a Staff member, I can choose the "Assist Customer" button to login in as that customer to provide him or her with excellent service.
  - Security Notes:
    - We need to have a ton of logging around this feature
    - Staff members should be required to have authenticated with two-factor so that we do not have an unauthorized person accessing this with just a staff credential

        - Let's identify certain private fields that customer service does not need access to while helping the customer. Those should be restricted; can we use the database SQL permissions to raise an exception if any of those fields is accessed while using this feature?

Examples of Misuse and Abuse Stories

When writing these requirements, remember that security is an emergent property of the system, not a feature. Start to think not as a normal user of the system, but as a user who is excessively curious at best or has malicious intentions at worst.

- As an authenticated customer, I see what looks like my account number in the URL, so I change it to another number to see what will happen.
- As an authenticated customer, I paste HTML that includes JavaScript into every field possible to see what happens.
- As a Malicious Hacker, I want to steal a copy of this database by any means possible, including by running Metasploit exploits against it's version of Ruby on Rails, obtaining a copy of the *database.yml* through a remote code execution, or by pretexting the cloud service provider pretending to be an employee at the company.
- As a Malicious Hacker, I want to gain access to this web application's Rackspace Cloud account so that I can lock out the legitimate owners and delete the servers and their backups, to destroy their entire business
- "As a Malicious Hacker I want to siphon credit card information so that I can use it for fraudulent purchases."[15]

Examples of Documenting Employee Misbehavior

One can also cover cases such as a team member ignoring best practices.

- As an outsourced junior developer, I have been asked to update the website with new content. To do this, I saved the password in a text file on my computer.
- As a developer, I want to be able to deploy a system easily so I commit session secrets to the Github.com repository instead of maintaining them in my own environment.
- As a developer, I have a copy of the web application and a clone of the production database on my laptop. I sometimes leave this laptop in my car unattended.
    - Security Note:
        - If a developer's laptop were to be stolen, legal tells us that we have a data breach unless the laptop is fully encrypted.

Practical technical countermeasures for your development team

In addition to the security concerns that are collaboratively documented as part of a user story writing process, developers should be familiar with the types of threats faced by a modern application. The OWASP Top 10[16] is an excellent start as are a variety of code analysis tools that are available depending upon the languages that the team is using. Each of these tools is best seen as part of a defense in depth strategy that starts while the system is being designed, with multiple countermeasures being put in place as the code is being developed, and then continues in deployment by the operations team. The team should also understand and make use of HTTP Security Headers[17], including Strict-Transport Security[18].

The role of a written information security policy

Many organizations do not have a written information security policy, but the presence of such a policy can greatly effect how the team goes about building custom software project and securing it in production. The policy need not be complicated, but it should state how the organization deals with sensitive information, such as formally adopting the information classification system presented in this paper. It should then go on to document what happens when a staff member leaves the company or when a system administrator is dismissed. It should cover the consequences to employees for mishandling customer information and it should include value statements that empower internal stakeholders to demand security be addressed as part of a custom software process.

Conclusion

Data breaches are a real concern today and organizations are legally and ethically obligated to protect their customer's private information that is entrusted to their care. Unfortunately, security is an emergent behavior that cannot be bolted on at the end through the use of SSL encryption or some network appliance. To have a chance at building a secure system, security must be discussed throughout the development process, starting with updating business processes to minimize the need for sensitive data in the first place. User stories with constraints and abuse stories are good tools for documenting these requirements collaboratively with the development team. During development, programmers have access to a variety of technical tools such as the OWASP Top 10 and HTTP Security Headers that are not language-dependent. These should be understood and used as appropriate. Finally, a written information security policy is an excellent way to codify an organization's view of security as part of its core mission and to ensure that security considerations are properly treated as requirements instead of being set aside as non-functional requirements.

---

[1] Verizon Enterprise. *2015 Data Breach Investigations Report*, pages 41 - 42. Retrieved 2015-08-10. Available: http://www.verizonenterprise.com/DBIR/2015
[2] Hoglund, Greg, and Gary McGraw. (2004) *Exploiting Software*, p 9. Addison-Wesley.
[3] U.S. Department Of Health And Human Services Administration for Children and Families. Information Memorandum. Retrieved 2015-09-01. Available: http://www.acf.hhs.gov/sites/default/files/cb/im1504.pdf
[4] Ponemon, Larry. (2015) *Cost of Data Breaches Rising Globally, Says '2015 Cost of a Data Breach Study: Global Analysis.'* Available: https://securityintelligence.com/cost-of-a-data-breach-2015.
[5] McCandless, David, et all. (2015) *World's Biggest Data Breaches: Selected losses greater than 30,000 records.* Available: http://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks. This interactive visualization is worth exploring, along with the raw data that's available in a Google Spreadsheet.
[6] Reader, Ruth. (2015) *The biggest lesson from Ashley Madison is about security, not fidelity.* Retrieved 2015-08-30. Available: https://venturebeat.com/2015/08/29/the-biggest-lesson-from-ashley-madison-is-about-security-not-fidelity.
[7] Northcutt, Stephen. (2009) *Security Controls. Retrieved 2015-08-31.* Available: https://www.sans.edu/research/security-laboratory/article/security-controls
[8] Rietta, Frank. (2014) *Commercial Information Security Classification System. Retrieved 2015-08-31.* Available: https://rietta.com/blog/2014/10/13/commercial-information-classifications.
[9] Official Code of Georgia Annotated. *Identity Theft, §§ 10-1-911*. Available: http://www.lexisnexis.com/hottopics/gacode.
[10] National Conference of State Legislatures. (2015) *Data Breach Notification Laws.* Available: http://www.ncsl.org/research/telecommunications-and-information-technology/security-breach-notification-laws.aspx.
[11] National Conference of State Legislatures. (2015) *Data Disposal Laws*. Available: http://www.ncsl.org/research/telecommunications-and-information-technology/data-disposal-laws.aspx.
[12] Cohn, Mike. *User Stories.* Available: http://www.mountaingoatsoftware.com/agile/user-stories
[13] Hope, Paco, McGraw, Anton. (2004) *Misuse and Abuse Cases: Getting Past the Positive.* IEEE Security & Privacy (Volume:2 , Issue: 3). Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1306981
[14] McGraw, Gary. (2004) *Software Security: Building Security In*, p 205. Addison-Wesley.
[15] Sterling, Chris. (2007) *Develop Architectural Needs Through Abuse User Stories*. Available: http://www.gettingagile.com/2007/09/16/develop-architectural-needs-through-abuse-user-stories.
[16] OWASP Foundation. (2013) *OWASP Top Ten Project.* Available: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[17] Queern, Caleb. (2013) *Test a Website's Headers for Security Features.* Available: https://securityheaders.com
[18] Lackey, Ryan. (2015) *Enforce Web Policy with HTTP Strict Transport Security (HSTS).* Available: https://blog.cloudflare.com/enforce-web-policy-with-hypertext-strict-transport-security-hsts/