Secure software delivery: who really owns it?

(Original creator: bolarotibi)

Guest contributors, Bola Rotibi and Ian Murphy from analyst firm Creative Intellect Consulting

Software breaches caused by flaws in software are meat and drink for the IT and national press. Barely a week goes by without another announcement of sites being hacked and passwords stolen. At the heart of this are software developers who, if the press is to be believed, are a bunch of incompetent nitwits. When we ran our Creative Intellect Consulting (CIC) 2011 and 2012 survey on “The Stat of Secure Application Lifecycle Management”, code development was viewed as the most susceptible for security bugs and issues. In 2013 the threat landscape has become more sophisticated and more targeted according to IBM’s 2013 mid-year results of the X-Force® Trend and Risk Report. But breaches are still happening as a result of vulnerabilities in the software code developed. But is it fair to blame the software developer? Anyone who has worked in software development will know how unfair much of the criticism is. Yes, developers make mistakes. Yes, this leads to software insecurity and enables hackers. No, not everything is down to the developer because software development is a not a single person issue. The CIC survey results indicated that developers have little influence in the organization when it comes to improving the security process. So whilst they are positioned to get the blame for problems, they have little power to make changes to the process to avoid those problems in the first place. Software companies are grasping at opportunities for revenue growth, often at the expense of security. However, the CIC survey also found that many roles, developers included, aren’t given sufficient guidance or process support to ameliorate the problem. Furthermore, resistance to addressing security in the software development lifecycle (SDLC) was found in the lack of support coming from developers’ managers rather than developers themselves. To challenge the charge that developers are most to blame for secure software failures, we hosted a secure development forum in partnership with ISC^2, a not for profit security education and certifying organization. At the forum developers and their managers from across a broad spread of industries and organizational sizes could make their defense. The outcome painted a very different picture of blame.

Collective responsibilities and culpability

Developers are not alone in the firing line. Unsurprisingly, the line-up is somewhat crowded when we examine the many different roles that play a hand in the failings with delivering secure software code, applications and systems.

Unrealistic commissioning clients

Given the number of times they were mentioned over the course of the forum discussion, end-user clients responsible for commissioning software applications must surely step forward. A lot of the issues highlighted focused on many end-user clients’ lack of understanding of the security implications, along with their inability to grasp the effort (financial through to time allocation) required to implement and deliver secure software. Perhaps more troubling was their willingness to assume security protection without specifically prescribing for it or detailing the acceptable levels of risks they were willing to undertake. Nor did this abdication of responsibility stop some clients from laying claim to “banking” level security protection even though few of them could guarantee it or confirm it through their own internal practices. When end-user clients outsource their application development and delivery requirements to small consultancies or agencies that specialize in specific types of applications e.g. Mobile device, social interactive web, or e-commerce apps, their security expectations remain unrealistic. Few are willing to invest sufficient financial resources into identifying or addressing potential security risks and threats. Most budgets don’t stretch beyond delivering the core functionality of the applications, and focus more on achieving the best graphical or digital experience that can be acquired for the sum. The suppliers know this. As a result it makes it hard for them to introduce security even when it is in the client’ s best interest to consider the risks and implications. Ultimately, if customers truly believe that they want “bank” grade security, five thousand pounds thrown in at the end for testing is not going to cut it. Unfortunately, many budgets assigned to external suppliers of digital content and applications are tuned for fast turnarounds and process “light” delivered solutions. They are set by line of business heads discouraged by what they perceive to be the slow pace of delivery from internal IT suppliers and over onerous implementation and execution processes and policies. As a result their cost expectations are at the significantly lower end of the scale, but their quality expectations markedly at the higher end. This disparity more than suggests an expectation for security but that it will be handled by those delivering the service within the budget assigned.

Upstream development phases are collectively letting the side down

Whilst code development was singled out from the results of the CIC survey as being the most susceptible phase for security bugs and issues to occur, it was closely followed by requirements capture and business process analysis and design as other key collection points. This naturally leads to the conclusion that many respondents believe that security issues arise from the way software applications are conceived and designed and with the creation of application code. But why should these upstream phases be the most prone to security bugs and issues? Code development makes sense because a coding error can have a significant impact on security. Security issues and concerns are sometimes overlooked when it comes to business process analysis and the requirements elicitation and capture process. This lack of consistently catering for and addressing security concerns at all stages of an application’s lifecycle goes to the heart of corporate governance and risk assessment and management. It poses a serious question as to who is responsible for software security. Is it the user, architect, designer, business analyst, developer, operations team, management, or the executive board? Ultimately it is everyone, especially as not every organization has the luxury of affording a dedicate information security personnel or teams to oversee security policy. What is required is for security and risk to be upmost in everyone’s mind and a first class citizen in everything that is done or required.

Security vulnerabilities don’t just lie in the code developed internally

A lot of the issues with insecure software actually come from the packaged, commercial software that enterprises buy. This software often has complex configuration routines and dependencies on operating systems that are outside the developer influence. Patching and maintenance of this software is an operational not a development responsibility. Other security issues are not software related at all. They are people failures where users have fallen prey to various scams and tricksters. This often leads to illegal software being installed on computers which then compromise the enterprise. In 2012, the IBM X-Force team reported that the top security breaches were often caused by known factors. SQL Injection is a well known security attack that has been in play for several years and is still top of the security attacks. Spear phishing where attacks use social media and other means to target a particular individual are also very common. Alongside this are malware which includes trojan software that allows attackers to take over computers. None of these are down to the developer. They are attacks that should be defeated by operations teams patching software and by security teams enforcing protection software.