Building an Effective Application Security Practice on a Shoestring Budget

Authors:
David Coffey, dcoffey@mcafee.com
John Viega, jviega@mcafee.com

Abstract:
Software companies inevitably produce insecure code. In 2006 alone, CERT has recognized over 8,000 published vulnerabilities in applications. Attackers were previously occupied by the weaker operating systems and have moved on to easier targets: applications. What makes this situation worse, is the weaponization of these exploits and the business drivers behind them. Some organizations struggle to deal with this trend to try to protect their products and customers. Other organizations have nothing in place, and need to create measures as soon as possible.

This whitepaper will raise several issues that global enterprise organizations currently face with application security and how to overcome them in a cost-effective manner. Some of the issues that will be discussed are software development lifecycle integration, global policy and compliance issues, necessary developer awareness and automated tools, and accurate metrics collection and tracking to measure the progress.

Table of Contents

Understanding the current threat

Companies and organizations live and understand how to perform risk management in almost every aspect of their business. Risk management has been a part of the business world relatively early on when people had to determine the risk of shipping their goods with one merchant over another. Today, the efficiency of risk analysis can be seen whenever you watch someone apply for a store credit card at the clerk's counter; the risk assessment on that individual is computed and decided within seconds. If we are so good at computing and assessing risk on an almost continual bases, why is it so difficult for a company to determine their current computer security risk?

The answer is in the unknown data, which ironically, is in their possession.

Computer security is a continually evolving field. Not only are new products and technologies coming out, but new attacks, tools, and protections are also being introduced all the time. "how do we keep the company safe?". This is the wrong question, the question that should be asked is "How safe is the company, where do we want to be, and how do we show we can get there?"

Computer Security threats are Evolving

The rule that most attackers will seek the "lowest hanging fruit" that will suit their needs is usually true. This has always been this way, from bank robbers grabbing money while it is being transported to decompiling java byte-code to obtain the static encryption key. The days of default passwords on the "postman" account are long gone (hopefully), and the days of application vulnerabilities are here.

If you have been exposed to any of the various computer security email lists, or news sources, it may be apparent to you that many of the issues being reported are in various applications and not the OS or network level. More than half of the FBI/SANS top 20 internet security attack targets are applications. This is a growing trend that many different people, including malware analysis. is starting to notice. The applications that we all know and love, coupled with the proliferation of the internet, are starting to bear fruit for the attackers; and they like it.

How did this happen? Lets take a look at the simple public-facing n-tiered web application. By the very design of this application's deployment, it is insecure; there is a custom written application that is running on expensive hardware which probably houses some private or sensitive data and any person with a public IP can mess around with it. There is no more perimeter or host authentication layer; it is just open to the public. Sure, there can be firewalls and IDS/IPS measures, but they are *designed* to let people through and not to hinder them. The application is designed to functionally perform as the sole line of defense in this situation, yet almost nobody takes this into account during the requirements, design, and development phase.

The cost of deploying an insecure product

Ultimately, the reason why companies develop applications is to provide a service to their customers, and while security is extremely important, so is the functionality that customers demand. When a company decides to deploy an insecure product, they are either knowingly doing it, or taking an un-educated gamble. The potential downside for releasing a vulnerable product is a public relations nightmare, angry and disappointed customers, a loss of revenue, possible lawsuits, and if it is a healthcare or military application, potental human casualties.

These issues almost always inter-relate to each other, with one either effecting or causing the other. Customers do rightly expect a certain quality from an application, and they definitely do not expect to be less secure by installing a product. The press particularly picks up on this fact and has no qualms with being as brutally honest as possible when discussing the issue in articles. This can lead to public embarrassment for the company which can result in a massive effort to not only effect a change in security posture but to show this change as well.

Security issues can lead to a loss in customer confidence and this may be reflected in everything from company stock price going down to loss in sales and signed contracts. Looking at the fallout of the recent TJX data security breach of 45.7 million stolen credit card numbers shows this to be quite evident where there is everything from a class action lawsuit to a share holder lawsuit. The

motivation of this issue is obvious where the attackers were able to obtain large numbers of credit card numbers, do it relatively risk free (they were doing it since 2002), and successfully use them (they had spent $1million USD since November of 2006).

The vendor's knowledge of security vulnerabilities falls into two unsurprising categories; they either know about them or they don't. The problem with the latter is that just because the vendor does not know about the issues does not mean that others are ignorant as well. There are many people that make a very good living off of finding vulnerabilities in applications belonging to other people. There are also people out there which are openly soliciting to pay for vulnerabilities as well. Both of these groups fall into the legal as well as black market areas.

Security researchers who work for companies help their company benefit from the bad PR they cause the other vendor. These companies hire teams of security researchers to find vulnerabilities in other vendor's products so that they can add the protection into their product and loudly tote that people are vulnerable and they have the only solution. This causes significant risk to the users of those "outed" products for relatively little and short-lived pay-off.

Other sources of vulnerabilities for a vendor comes from internal employees and customers. Occasionally a customer notices something strange, or a customer's internal security team catches an issue in one of their reviews. When these are discovered, as with vulnerabilities discovered externally to the company, it is important to have a process in place to deal with them. Most researchers will adhere to some sort of disclosure policy, and they should be able to communicate this policy when asked. For constructing a vulnerability remediation policy, refer to the Organization for Internet Safety and the National Infrastructure Advisory Council for more information.

Regulations and Compliance

Application security is playing an increasing part in new regulatory and compliance controls being placed on companies. Traditionally this has played a part only in the government space, but it is moving into financial,

health, and traditional businesses. These range from US laws like the Sarbanes-Oxley Act to international standards like ISO 27001 and ISO 17799.

The main reason for the Sarbanes-Oxley Act (SOX) is to prevent corporate fraud. It was signed in 2002 and companies only have to adhere to it if they are publicly traded and have a minimum revenue of $75 million. The sections that pertain to application security all deal with enforcing the integrity, confidentiality, and availability of data within a company. The specific sections we are discussing here are 302, 404, and 409 within the Act. Ultimately, as stated very clearly in section 302 of this Act, corporate officers are to be held accountable to the accuracy of the final report and hence the internal controls and accuracy of the data within the company. This can very quickly be translated into data breaches that allow manipulation of data will be taken very seriously and should mitigated at all cost. By holding the executive level accountable, it is in the direct interest of the company and the officers, to adhere to this law.

The ISO 17799 started off as a direct copy from the British Standard 7799:1995 in 2000, and has been revised since then. The standard serves as a reference of best practices for information security management within an organization. It is comprised of a series of best-practices and is widely accepted around the world. The sections of this standard that deal primarily with application security are parts of section 10 (10.4, 10.6, 10.9, 10.10), 11, 12, 13, and 15. These sections cover network communication, logging, access control, development and maintenance, incident management and compliance respectfully. While it might not dictate exactly how to secure a specific type of network, it does offer sage advice for the security around outsourcing and access control.

The Gramm Leach Bliley Act (A.K.A. GLBA, GLB, and “Financial Modernization Act of 1999”) was put into effect in 1999 and title V is designed to prevent identity theft. It does this through detailing what customer data can be collected, how it can be stored, and how it can be disclosed. This law is enforceable on all “financial institutions”, which can allow a law enforcement agency to interpret it as any organization that deals with the financial data of an individual and not just banks. With applications that need to adhere to this act, all

applications written to run in a "financial institution", the functionality must be present that enforces data protection, confidentiality, and integrity throughout the system. Interestingly, there is a section that applies almost exclusively to social engineering, which adds a new dynamic to coning people out of financial data.

The Visa Payment Card Industry Data Security Standard (PCI standard or PCI DSS) was established in 2001 as part of the VISA Cardholder Information Security Plan. The purpose of the PCI standard is to establish a set of rules and guidelines for what type of data a merchant can collect, store, and how to protect it. If an online merchant is to be PCI compliant, they need to be validated on a quarterly basis by an approved PCI security vendor. Applications which need to adhere to this standard not only need to worry about what is being stored, but how they are being stored as well as being resistant to specific vulnerabilities.

Regulations are an interesting catalyst in the application security field. Most of these laws have not been established for 10 years yet, and are having a dramatic effect on the state of security within organizations. Even if they only serve as a checkbox, a formality that must be accomplished, the organization is doing something to increase the security of their systems and data. The problem with these regulations is that they dictate what must be secure, but not how to secure it.

How to get secure

Knowing where you need to get to is usually relatively easy when compared with the process of actually reaching your destination. The same can be said about application security, it is easy to make the claim that your applications must be secure and very difficult to actually make them so. There have been several books written about this topic of implementing application security best practice areas, so we will only briefly cover it here.

Ultimately, the best practices break down into four areas, which are:

- Application Security Awareness
- Security Assessment
- Process Integration and Automation
- Assistance and Mentoring

Application security awareness among both developers and management staff is crucial to any initiative. If developers are unaware of the issues that can result through insecure code and managers are not in support of the effort, it will die before it begins. Awareness of these issues is best accomplished through some sort of education process. Developers will usually learn best by attending a technical application security course. This course can be generated in-house, but it is suggested that outside help is elicited to obtain that degree of specialization and authority on the subject. If money is an issue, a good selection of books will work well or the local college may offer or willing to start a secure programming course. Several technical conferences also exist that can provide good exposure to new techniques and methods.

Assessment of the current state of application security can be done by any combination of a penetration test, threat model, and source code audit. Any one of these will quickly expose underlying issues that were previously not known or thought not to be too serious. It is suggested that when first performing these to again seek outside assistance as the quality of results can be greatly enhanced by leveraging the specialization and experience of a seasoned individual. Regardless, performing these activities through a small group of developers will be beneficial and can often times be enjoyable by the team. Not only do you end up with a list of security issues, but members of the assessment team better understand the product and how the pieces interact.

Process assessment is also equally important, especially when determining how best to integrate application security activities into the Software Development Lifecycle (SDLC). There are several activities that can be integrated here, like security requirements, code reviews, threat modeling, and security QA testing. It is suggested that the organization pick and choose which activities will work best in their process, and integrate them piecemeal. Application security does not have to happen all at once, and it is better to be doing what you can rather than nothing. Once the activities are started, generating worksheets, howtos, and even brown-bag classes on it will help greatly with internal buy-in and adoption.

Automation is another piece of the process integration, and works best when it is mandatory and un-avoidable. There are several technologies which can assist an organization with automating security testing and assessment of specific areas. These range from automated black box testing and scanning to source code static analysis tools. These are usually off the shelf products, and it is suggested that a "bake off" or similar proving of the technology is performed before a purchase is made. Regardless, integrating the black-box and scanner technologies into the build and QA process will enable engineers to see early issues faster and on a consistent basis.

Providing application security assistance to the developers and management within an organization is absolutely crucial to make sure that things are keeping on track and issues are fixed properly. Developers should not be expected to be become experts in the application security field. Their job is ultimately to translate obscure product requirements into a functions product, not to worry if it is being the most secure as possible. Providing this service through a specialized role will help issues be resolved cleaner and keep the project moving smoother. Sometimes this role can be filled by a security conscious developer, but in application security, there is no substitute for experience.

These best practice areas have been around for a while now, formalized for at least 7 years. One thing that has not been discussed much is how to best show the return on investment in the program. This is a relatively new field, and has partly established itself because of new laws and regulations.

Showing the effectiveness through numbers

The computer security division in most companies often have issues with being viewed as a "cost center" along with not being able to show the effectiveness of their actions. This is usually due to a combination of the business not understanding and the security division's failure to properly communicate and track their actions. If the security division is doing the job properly, then nobody should know they exist because there is a lack of noticeable issues. This leads to budget cuts and downsizing of the security team, which has a crippling effect on a company. When this happens, the management

team of the security group needs to revise the operational procedures. There are ways to show a Return on Security Investment (RSI), and the numbers are already available to people.

When dealing with metrics, you can only measure what you know. Instead of asking "How do I show that something is secure?" try asking "How do I know this is more secure than it was?". Trying to prove the former of the two questions will result in frustration and loss of time. There is usually a lot of data that can be tracked, and with creative applications, often lead to astounding revelations. If properly done, not only can it show the security group as preventing loss and having direct monetary benefit, but also enable budget increase along with new projects that will further enhance security.

Some of the metric points that can be gathered are:

- Number of bugs and severity resulting from a source code audit
- Number of lines audited
- Number of security bugs reported in by researchers, how long it took to fix them
- How many people have been trained and in what
- How many whitepapers, policies, documentation have been developed

Though these are fairly simple points to record, combinations of them show some powerful results. Tracking the number of bugs and severity that have been discovered as part of a code audit along with the number of lines audited will give you the density of defects. Breaking this out by vulnerability type and product will allow for trends to be established over time. This can also result in the establishment of the most critical flaws that need to be addressed within the organization and if it is common across different projects.

If security researchers report issues, tracking these will not only show what the researchers believe to be the most critical areas in your applications, but it will also allow you to match these up to your source code audits. If you have discovered the issues in audits, and are proceeding to fix them, you have been doing the job effectively. If the product was audited and the issues were not found, then the code audits are obviously deficient in this area and should be resolved if possible. These numbers will either show

that the job is being done efficiently or it will show a deficiency and how to proceed with fixing it. Both of these are good things.

Training is another metric that is often overlooked. If you know which groups and who is being trained, then the effectiveness of the training can be measured through automatic and manual code analysis. If training has occurred, and static analysis results show a downward trend in discovered issues, the training has been demonstratively effective. If results are consistent in a subset of areas, then you know which parts of the training need to be strengthened. With the coupling of source code control and automated static analysis, it is also possible to show these trends on a per-developer basis. This will not only show which developer is the most security conscious, but also which developers were effected more by the training.

Performing analysis like this allows goals to be set for the security division that align with the organization's business goals. Establishing a 20% reduction in reported vulnerabilities is not only tractable, but by leveraging the other metrics, provably achievable.

Conclusion

Overall, with the establishment of logical activities and measurements, an application security practice can not only be created, but shown to be effective. It is often a requirement to build such a team with a tight budget, but the better things perform, the better the group will be viewed within the company. It is often difficult to escape the association with bottleneck and inefficiencies, but by providing strong metrics and aligning with the business needs, a security group can find themselves not just surviving but thriving.

References

Unknown Authors, SANS/FBI top 20 Internet Security Attack Targets, SANS institute, 2007 (http://www.sans.org/top20/)

Unknown Authors, Computer Emergency Response Team Statistics, CERT, 2007 (http://www.cert.org/stats/)

Bill Brenner, Banks prepare lawsuit over TJX data breach, SearchSecurity.com, April 2007, (http://searchsecurity.techtarget.com/originalContent/0,289142,sid14_gci1252778,00.html)

Unknown Authors, Organization for Internet Safety Guidelines for Security Vulnerability Reporting and Response V2.0, Organizatin for Internet Safety, 2004, (http://www.oisafety.org/guidelines/secresp.html)

Unknown Authors, Vulnerability Disclosure Framework, National Infrastructure Advisory Council, 2004, (http://www.dhs.gov/xprevprot/committees/editorial_0353.shtm)

Findlaw.com posting of "The Sarbanes-Oxley Act of 2002", 2002, (http://news.findlaw.com/hdocs/docs/gwbush/sarbanesoxley072302.pdf)

Senate Banking Committee posting of Gramm-Leach-Blily Act "Title V – Privacy – Subtitle A – Disclosure of Nonpublic Personal Information", November 1, 1999, (http://www.senate.gov/~banking/conf/fintl5.pdf)

PCI Security Standards Council posting of "Payment Card Industry Data Security Standard", 2007, (https://www.pcisecuritystandards.org/tech/download_the_pci_dss.htm)

British Standards Institute (BSI), "BS ISO/IEC 17799:2005", 2005, (http://www.iso17799.net/)

John Viega and Gary McGraw, Building Secure Software: How to Avoid Security Problems the Right Way, Addison-Wesley Professional,Septermber 24, 2001