FEBRUARY 2002

The Injustice of Insecure Software

By
Abner Germanow
Chris Wysopal
Dr. Dan Geer
Chris Darby
research@atstake.com

Abner Germanow the Services Marketing Manager with @stake, Inc, Previous to @stake, he was the Research Manager for Internet Security at the market research firm IDC.

Chris Wysopal is the Director of Research and Development at @stake. He is one of the primary authors of the standards for public vulnerability disclosure currently before the IETF.

Dr. Daniel Geer is the Chief Technology Officer at @stake as well as the current president of Usenix.

Chris Darby is the CEO of @stake. He frequently lobbies for secure software initiatives to government policy makers and in other public forums.

Software vendors often profess the importance of security. History, however, suggests otherwise: the software market has failed to produce secure software. This paper reviews the root cause of this market failure, risks this failure exacerbates, myths of today's solution, and @stake's solutions to remedy the crisis.

The Market Failure

Customers demand software that reliably delivers a desired set of functionality. The competitive race to satisfy this demand for functionality drives software profits and the speed of the software development lifecycle. Participating in this race requires vendors to balance product perfection against competitive advantage and income statements. This precarious balance produces the release of software filled with flaws.

Flaws, where an application fails or conflicts with other software, can range in severity from mundane annoyances to security vulnerabilities that place national infrastructures at risk. While mundane annoyances are a low price to pay for competitive advantage, the risks users are willing to accept in exchange for functionality is reaching a critical inflection point. Software has become too important to the livelihood of too many organizations to regard the repetitive discovery of well known and preventable security flaws to be accepted as "normal." [1]

The daily volume of well-documented security vulnerabilities in mainstream software suggests the technology market is failing to learn from past mistakes and failing to serve the common good. It is high time for change.

Growing Risks

@stake estimates 30 to 50% of the digital risks facing IT infrastructures are due to flaws in commercial and custom software.[2] Critical infrastructures, meaning the ones delivering electronic services in a networked world, require security across people, processes, and technology. The state of the software industry today places technology users at tremendous risk. These risks can be grouped into three areas:

- **Transferred Risk:** Risk a customer takes by using a commercial product. The customer must rely on the vendor to produce secure software. If a security vulnerability exists in a piece of software and the customer cannot mitigate it through configuration changes, then the vendor has *transferred the risks in the*

product to the customer. Since the customer controls neither the design nor the implementation of the software, the burden of fixing the product falls on the vendor.

- **Owned Risk:** Risks a company takes by building a solution in-house. If a security vulnerability is discovered, *it is the company's own responsibility* to address the problem. Since the company owns both the design and its implementation; there is no vendor to rely upon. The complexities of configuring many products tend to blur the line between transferred risk and owned risk.

- **Uncertainty:** Companies face risks *regardless of the technology or solution* deployed. Since no solution is ever 100% secure, the company will be well advised to limit the probability of cascade failure (where one failure triggers another). Minimizing cascade failure begins with emergency or contingency procedures that can react to unknowable scenarios, and is the principle connection between security and reliability.

The language of software license contracts has always absolved software vendors from incurring liability from the risks they transfer to their customers by releasing insecure software. @stake (and others) believe it is only a matter of time before a major software vendor or service provider is held liable for failing to address the security in a product or service. In the meantime, truly "leading" vendors are the ones taking steps to reduce these risks today. On average, the best applications reduce business risk 80% more than the worst.[3]

The Goal of the Digital Attacker: Become An Insider

Other than a simple vandal, an external attacker's the first measure of success is to gain the credentials of internal, legitimate users. Whether the attacker takes over a global trading network or a single account in a supply chain system, "intrusion detection" is over once the attacker has become an insider. Today, the path of least resistance to this goal is not the network gear, not crypto-mathematics, nor bribery; it is application software.

The value of secure software

Even in a world clamoring for security, the reality of corporate budgets and the necessity of productivity gains dictate that security is rarely an end goal unto itself. Instead, security is a means to achieve other goals such as: software quality, reliability, flexibility, reputation protection, and services trustworthy enough to provide critical services.

Quality in today's software industry answers the question, "Does this software do what it is supposed to do?" Reliability means, "Does this software do what it is supposed to under stress?" Software vendors who care about risk transfer are expanding traditional quality assurance efforts with security tools and techniques that answer the question "What can this software do that it should not do?" Secure software might not be perfect, but is trustworthy and high quality.

Why security flaws persist

Competitive business pressures accelerate in a shrinking world and software has a lifecycle, a turnover time, sufficiently short to meet many of those pressures. For software suppliers, there is an unquenchable consumer demand for features, yet absent the most serious of attentions, expanding functionality typically expands complexity. And complexity is the enemy of security beyond all others.

Economic data from the 1990's shows that the acceleration of product cycles resulted in the near doubling of annual productivity gains in that period for advanced societies, so the simple suggestion to heed "Haste makes waste" is itself simplistic.[4] With this pressure, an uneducated market is likely to demand answers to security that can be purchased with a check on Monday and delivered by a truck on Wednesday. Conversely, an educated market understands that true security cannot be boxed and sold, but rather must be part of every process. Secure software engineering obviously contributes to the security of the product, but more importantly contributes to the quality of the product. Educated consumers are now demanding a level of quality in software, network services, and web services that are *secure at release*.

Given the importance most software vendors profess to give security, business realities have traditionally prioritized design agendas with customer experience and increased functionality. Software vendors often view software quality and security as two separate issues. For example, in 2001 software vendors spent more than $8.4 billion on application design and construction tools,[5] while total spending on application security tools, training, and consulting was less than $50 million in 2001.[6]

One might also expect security flaws to be a consequence of rushed programming, accidents, or incompetent deployment. This is false. @stake has found 70% of security vulnerabilities in software are a result of design flaws no tool could single handedly prevent.

Solutions

What progress has the market made toward secure software?

Five years ago, software vendors shuddered at demands to fix security flaws in their products. Researchers, many of whom are now members of the @stake team, had no effective choice but to publish the existence of a vulnerability and often even write and release an exploit if they were to have any hope of a vendor fixing the problem – frontier warfare is rarely beautiful. The pressure these pioneers generated has evolved into the penetrate-and-patch world of today. Patching will never go away, but the days of @stake or anyone else needing to publish an exploit to encourage a vendor to fix a problem are long over. Most vendors now have some sort of reaction-team to issue patches with minimum latency just as most mission critical IT operations have teams to test and deploy that constant stream of patches. This model alone cannot solve the problem.

The myth of today's solution

Patching is a short-term fix, and not a very good one. Obviously, many environments would be more secure if they consistently installed the latest patches. However, there is now real data proving that even patched corporate infrastructures are a body of risk that cannot stand – something has to give.[7] Patching by nature requires a development time that forces software consumers to accept the burden of a constant window of risk between "good guy" discovery of the problem and patch deployment. Even worse is the larger and fuzzier window of risk between "bad guy" discovery and "good guy" discovery. History shows that a small but important group of vulnerabilities will be exploited before the software maintainer knows about the problem.

Security via patching is expensive, time consuming, often unproductive and never on time. Don't believe any claim that automated patching would be a path to security nirvana. Effectiveness of patches is somewhere between band-aids and a stiff drink.

Investing in secure software development

The return on investment for integrating secure software engineering principles into the development cycle is still an inexact science, mostly because of the difficulty of data acquisition. Regardless, the magnifying glass has turned to software security and leading vendors have already begun to examine and shift the development process. If @stake is correct in predicting near-term changes in liability assignment for software flaws of global reach, some combination of regulatory pressure, insurance-based risk transfer, and straight P&L considerations will soon sharpen up the data on which ROI decisions will be made.

By addressing questions of security in the design and testing stages of software development, security engineering has the substantial advantage of the massive, pre-existing literature on quality assurance on which to lean, from which to adapt, and to which to contribute. While the last twenty years of quality studies have synthesized conclusions in a variety of ways from "Quality is Free" to General Electric's Six Sigma methodology, the research is conclusive. Investments in quality yield attractive returns when the consequences of poor quality are fully costed. Quality and security are synonymous.

@stake's secure software mission

The security professionals on the @stake team pioneered the efforts to convince, cajole, and even embarrass software vendors into mitigating exposed risks. That was a good start, but there is much more to be done.

@stake, as always, is committed to informing our clients and colleagues of digital risks as early as it is feasible to do so. More importantly, we are committed to making it easier for software vendors to achieve *secure software at release*. This commitment is perhaps best demonstrated through the following initiatives:

1. **Application Services, Tools, and Education:** @stake is the leader in helping software developers design, implement, and test secure applications. SmartRisk services, @stake Academy courses, and tools help software vendors[8] develop

more secure products. @stake has expended significant resources in advancing secure software engineering methodologies, courses, and developing tools where none have existed.

2. **Return On Security Investment (ROSI)**: Security investments must compete for budgets and thus must demonstrate well-articulated financial arguments. @stake's SmartRisk Services can quantitatively measure returns on security investments and otherwise rationally connect your CFO, CISO, and CIO in the decision making process.

3. **Standards for public vulnerability disclosure:** Patching, discredited or not, will continue to be an integral part of digital security for some time. For anyone involved, to perform at "best practices" standards requires standards – what a surprise. Global IT operators, software vendors, and security professionals all have an interest in standardizing procedures for security flaw handling. This is not a time for commercial posturing nor for conspiracy theories about the motives of those who are at least working on a solution. Much like the refrain "Vote or don't complain," @stake requests every reader of this piece to help with this fundamental question: How can we make security flaw information propagate while simultaneously minimizing collateral damage and patch latency *in the installed base*?

4. **Legislation and the public sector**: @stake is actively involved in efforts to:

    - Solidify the ability of our clients to examine the security of the products they purchase and deploy.
    - Make investment in security economically advantageous outside any market failures that might otherwise be said to intrude.
    - Contribute our own staff to the protections of critical infrastructures both public and private.

The Bottom Line

Secure software at release is the end game; @stake has always and will always fight for the security of our clients and the software we all depend on. @stake dragged the market to the penetrate and patch state of today and will now push the market to justifying appropriate investments in security. Just as the quality assurance literature showed before, early intervention during design is far more effective than remediation, and @stake has the numbers to prove it.[9] As soon as this sinks in widely, or sooner if events force a general reassessment of security liabilities, the quantitative analysis of security in an ROSI context will be the norm, not an outlier.

Bringing the entire software market to *secure at release* will be a long battle, with few obvious victories. Victories will come slowly over time as customers demand quality software, the volume of security patches decline, and the software supporting critical infrastructures and commerce does its job quietly and safely.

About @stake, Inc.

@stake provides corporations with digital security services that secure critical infrastructure and electronic relationships. @stake applies industry expertise and pioneering research to design and build secure business solutions. As the first company to develop an empirical model measuring the Return On Security Investment (ROSI), @stake works where security and business intersect. Headquartered in Cambridge, MA, @stake has offices in Denver, London, New York, Raleigh, San Francisco, and Seattle. For more information, go to www.atstake.com.

Notes and references

[1] Computer Science and Telecommunications Board. "Cybersecurity Today and Tomorrow: Pay Now or Pay Later," *National Academy Press*, Washington, D.C., January 2002.

[2] @stake estimate based on anecdotal evidence from multiple digital forensics engagements along with attack data from the 2001 FBI/CSI statistics

[3] A. Jaquith. "Application Security: Not All Are Created Equal," *@stake Research Report*, February 2002.

[4] Dale W. Jorgenson. "Information Technology and the U.S. Economy," *Presidential Address to the American Economic Association*, New Orleans, Louisiana, January 2001.

[5] Rikki Kirzner. "Revised Application Design and Construction Tools Market Forecast and Analysis Summary, 2001-2005" *IDC*, November 2001.

[6] @stake estimate based on @stake and competitor revenues

[7] William A. Arbaugh, William L. Fithen, and John McHugh. "Windows of Vulnerability: A Case Study Analysis," *IEEE Computer*, December 2000.

[8] A sampling of vendors @stake has partnered with to develop secure software can be found at: http://www.stake.com/services/clients.html

[9] K. Soo Hoo, A. Sudbury, A. Jaquith. "Tangible ROI Through Secure Software Engineering," *Secure Business Quarterly*, Q4 2001.