Interview

We're up against

Gary McGraw

Gary McGraw, Cigital, Inc.'s CTO, is a world authority on software security. Dr. McGraw is co-author of five best selling books: Exploiting Software (Addison-Wesley, 2004), Building Secure Software (Addison-Wesley, 2001) Software Fault Injection (Wiley, 1998), Securing Java and Java Security (Wiley, 1996). His new book Software Security: Building Security In (Addison-Wesley) was released in February 2006.

hakin9 team: Would you please introduce yourself, tell us about your background in the security industry, and remind what is Cigital?

Dr. Gary McGraw: Sure. I am Gary McGraw, CTO of the software quality firm Cigital *www.cigital.com*. Cigital is a consulting firm in the United States that specializes in helping software producers build better software. In particular, we focus on software security and software reliability.

I got started in the security field back in 1995 when I joined Cigital (at the time called Reliable Software Technologies) as a research scientist. I have a PhD from Indiana University in Computer Science and Cognitive Science where my advisor was Douglas Hofstadter. Part of my job at RST was to research whether software fault injection would be a useful technology for security. At the time, we were applying software fault injection to safety-critical systems, and we wanted to see how far it could be pushed in security. Eventually I wrote a book about that technology called *Software Fault Injection*.

During the same time period, I became very interested in Java and Java security. We downloaded Java when it was still in alpha and started playing with it. As a programming languages guy, I was particularly interested in Sun's security claims. What does it mean for a language to be secure? How did the Java security model really work? I got together with Ed Felten from Princeton and we wrote Java Security, in which we described the many ways in which we had broken the Java Virtual Machine.

In 2001 I wrote with John Viega *Building Secure Software*. That book set off a revolution in computer security, and helped to jump start the field of software security and application security. I followed *BSS* up with *Exploiting Software*, a book on breaking software co-authored with Greg Hoglund.

My latest book *Software Security: Building Security In* (*www.swsec.com*) was released this year. This book, which talks about how to DO software security, describes a set of seven software security touchpoints that all developers should adopt. The top two touchpoints, each of which gets a chapter, are code review with a tool and architectural risk analysis.

I'm working on a new book with Greg Hoglund now. I also write a monthly column for *www.darkreading.com* and host a podcast called the Silver Bullet Security Podcast with Gary McGraw *www.cigital.com/silverbullet*.

h9: What do you think about the situation on IT security scene? Do you think it's developing in the right direction?

GM: I am optimistic that we're making progress. Let me clarify that – I don't think much progress has been made in network security for quite some time, but the advent and rapid growth of software security is great. So as a whole, we're making progress since software security is coming along nicely.

Ten years ago when I started talking about software security, everybody thought I was crazy. They all thought that security was about firewalls, intrusion detection systems, and anti-virus. Today, everyone seems to realize that we have a serious software problem and we need to gear up to address it.

My books have evolved along with the field, moving from philosophy and problem description in *Building Secure Software* through explanations of how things really break in *Exploiting Software* all the way to what we need to do about bad software in Software Security. All three books have been packaged together into a boxed set called the Software Security Library (*www.buildingsecurityin.com*).

I truly believe that the only way we can begin to make forward progress in computer security is to focus more attention proactively on better BUILDING and much less on reactive solutions like firewalls. That means communicating with developers and engineers. So far, we seem to be making steady slow progress.

h9: Please say what you think are the success factors for a security-oriented products?

GM: I think most security products are awful, actually. Firewalls don't do as much good as people think. Intrusion detection systems are basically noise makers. Anti-virus solutions react to software exploits only after they have been propagated. Patch management systems are designed by people who think we can patch our way out of the software problem that we have.

Heck, even in software security we have our share of snake oil products. Early hacker in a box application security testing tools are no better than badness-ometers. That is, they can show you in no uncertain terms that your software is terrible, but they can't show you that it is secure. And application firewalls are about the silliest idea ever. I suppose they are marginally useful if you didn't build the software you are protecting. But if you did, these kinds of checks should be in the code not at the network level in some pizza box.

By contrast, I am pleased with the advent of software security tools like static analysis tools for code review. I had a hand in bringing the Fortify toolset to market, and I am pleased with what that company is doing *www.fortifysoftware.com*. Tools for builders and testers seem to me to be the next big market in security. I want to make sure that the tools are actually done right.

h9: Computers are everywhere, perhaps that thesis is trivial, but do you think that home users are aware of the danger? How to protect your system being home user only, not spending large sum of money, we don't have actually, on security tools? Are tools available on the Net valuable?

GM: I think clueless home users are a big problem (just look at botnets), but that the problem is causet by operating systems vendors (like Microsoft), who have only recently begun to take security seriously. The good news is that Microsoft cares about software security. The bad news is that it will take years and years to fix the problems.

Home users are in a quandary today. They are forced to buy extra security products if they want their machines to be secure. Ironically, Microsoft is entering this market, promising to deliver software that will protect you from the risk caused by their other software! What a scam! I rely on off-the-shelf commercial products for my own pile of PCs. I use Norton Internet Security. I find it valuable enough to pay for.

h9: During past years, we've seen record breaking reported vulnerabilities. Could you briefly present your thoughts on this situation? What do you think is the primary reason? What is the biggest problem of network security now and in the future?

GM: You know what I am going to say already! The biggest reason that we have a huge and growing computer security problem is because of broken software. Thought the widespread adoption of network security technologies continues, the problem persists. The data from 2005 are even worse, with the number of vulnerabilities going up again. The biggest problem in network security is software security.

h9: Thought or at least positioned to be secure products have started putting a lot of efforts to patch the numerous vulnerabilities that keep on getting reported. Is it the design of the software itself or the successful mass patching and early response procedures that matters most in these cases? There are some problematic questions connected with security. I'm wondering who is responsible for vulnerabilities in the system? Who should be punished, if anybody?

GM: It's pretty funny that security product vendors don't really practice software security. Their products are as riddled with security vulnerabilities as any other set of products. You see, security software is not software security! That's a subtle but important lesson to internalize.

Good design and good implementation are much more important than some kind of patching regiment. Penetrate and patch is a terrible idea. We will never completely eradicate patching (because we need it), but we certainly can't rely on patching to secure our broken software. My new book is all about what you should do in the Software Development Lifecycle (SDLC) to avoid having to patch later.

Today it is not clear that anybody gets in trouble when software is shown to be insecure. I am not a fan of imposing personal liability on developers (as some crazy pundits have suggested we do), but I do think that software producers need to be held more accountable for their successes and failures when it comes to security.

I believe that the Market itself is starting to ask better questions about software security. This is in turn causing vendors (including Microsoft) to address software security head on. As a free market capitalist, I think the Market is working properly in this case.

h9: What do you think should be done for security in day-to-day life? Do you think beta tests influence on applications quality?

GM: Do you mean physical security or computer security? I'll just assume that you mean the latter. I suppose if I were a politician I would have to focus on terrorism or some other such minor risk!

In terms of software, you should ask hard questions about what the software vendors you are relying on are doing about software security. Ask them what they did to secure their software. Make them show you analysis results. This goes a long way to determining whether they have a clue (and hence whether you should use their stuff).

If a vendor says: *everything is secure, because we distribute only binary versions of our software*, you know they are idiots. If they say, *everything is secure because we use SSL*, you know they have their hearts in the right place, but they are confused. If they say, *we had a security audit of our software performed by a trusted third party and you can talk to them*, you know they are at the cutting edge of software security.

Beta testing is a bad place to try to measure and enhance quality. If a software vendor is relying on customers to replace their professional QA staff, they are likely to be producing lousy software. So, waiting to assess your security posture in beta is crazy!

On the other hand, both security testing and penetration testing are important software security best practices. They go hand in hand with code review and architectural risk analysis.

h9: Why, in your opinion, security is still problematic question for programmers? How to build secure software? Is it possible?

GM: Of course it is possible! That is, making software 100% secure is not possible, but properly managing risks in software so that it is secure enough is definitely possible.

In *Software Security*, I introduce a risk management framework that is very helpful when applying the software security touchpoints. By using a risk-based approach, you can ensure that software security is applied in a sane fashion.

Security is problematic for developers for a number of reasons. First of all, most developers have never been taught anything about security (even network security). They think security is somebody else's problem. Second, even if they do realize the importance of security, they have a natural propensity to focus on security features (like cryptography) instead of security vulnerabilities (remember, software security is not security software!). And third, developers have come to be very wary of security people, mostly because security people occasionally show up with sticks and beat them senseless for no reason.

We have to get beyond these three problems by adopting the software security touchpoints described in *Software Security*. If you know any developers, get them a copy of the book, and make them read it!

h9: What do you think about commercial and open source applications security?

GM: Viega and I have a discussion about this in *Building Secure Software*. My position really has not changed since then. Both proprietary (or commercial) software and open source software need better software security. From an economic perspective, proprietary software is probably in a better position, because enterprises can pay for assurance work. Open source projects must rely on volunteers.

In either case though, all of the software security touchpoints should be applied.

h9: In your black and white hat books you present mirror images of software security, where firewalls, antiviruses and other tools seems to be not good enough. What is, in your opinion, a tool that gives users the best security nowadays? What security products do you use, what could you recommend? Where is the balance between attack and defense?

GM: The black and white hats are symbolic of the need for both attack and defense in security. In the book preface, I say: *the yin/yang design is the classic Eastern symbol used to describe the inextricable mixing of standard Western polemics (black/white, good/evil, heaven/hell, create/destroy, and so on). Eastern philosophies are described as holistic because they teach that reality combines polemics in such a way that one pole cannot be sundered from the other. In the case of software security, two distinct threads – black hat activities and white hat activities (offense/defense, construction/destruction) – intertwine to make up software security. A holistic approach, combining yin and yang (mixing black hat and white hat approaches), is required.*

Finding a balance is tricky, but it is clear that neither all offense nor all defense will work as approaches. I believe in the use of technology and tools in support of both black hat and white hat activities.

h9: What do you think about hackers community? Is it connected (I mean ethical hacking), with new security ideas and improvements?

GM: It is essential that we understand what we're up against. For that reason, I have never shied away from talking explicitly about software attacks and how they work. I wrote *Exploiting Software* with Greg Hoglund (who runs *rootkit.com*) for just that reason. I believe we need to understand the attacker's toolkit and how it is wielded. I wanted people to understand more about how software breaks and what people do to break software.

As for people who carry out illegal or malicious hacking, I think they should be punished like any other criminals when they get caught. ●

Interviewed by Marta Ogonek