Anonymous speaks: the inside story of the HBGary hack

By Peter Bright | Published 7 months ago

It has been an embarrassing week for security firm HBGary and its HBGary Federal offshoot. HBGary Federal CEO Aaron Barr thought he had unmasked the hacker hordes of Anonymous and was preparing to name and shame those responsible for co-ordinating the group's actions, including the denial-of-service attacks that hit MasterCard, Visa, and other perceived enemies of WikiLeaks late last year.

When Barr told one of those he believed to be an Anonymous ringleader about his forthcoming exposé, the Anonymous response was swift and humiliating. HBGary's servers were broken into, its e-mails pillaged and published to the world, its data destroyed, and its website defaced. As an added bonus, a second site owned and operated by Greg Hoglund, owner of HBGary, was taken offline and the user registration database published.

Over the last week, I've talked to some of those who participated in the HBGary hack to learn in detail how they penetrated HBGary's defenses and gave the company such a stunning black eye—and what the HBGary example means for the rest of us mere mortals who use the Internet.

Anonymous: more than kids

The HBGary saga:

Anonymous to security firm working with FBI: "You've angered the hive"

How one security firm tracked down Anonymous—and paid a heavy price

(Virtually) face to face: how Aaron Barr revealed himself to Anonymous

Spy games: Inside the convoluted plot to bring down WikiLeaks

Anonymous speaks: the inside story of the HBGary hack

Black ops: How HBGary wrote backdoors for the government

HBGary and HBGary Federal position themselves as experts in computer security. The companies offer both software and services to both the public and private sectors. On the software side, HBGary has a range of computer forensics and malware analysis tools to enable the detection, isolation, and analysis of worms, viruses, and trojans. On the services side, it offers expertise in implementing intrusion detection systems and secure networking, and performs vulnerability assessment and penetration testing of systems and software. A variety of three letter agencies, including the NSA, appeared to be in regular contact with the HBGary companies, as did Interpol, and HBGary also worked with well-known security firm McAfee. At one time, even Apple expressed an interest in the company's products or services.

Greg Hoglund's rootkit.com is a respected resource for discussion and analysis of rootkits (software that tampers with operating systems at a low level to evade detection) and related technology; over the years, his site has been targeted by disgruntled hackers aggrieved that their wares have been discussed, dissected, and often disparaged as badly written bits of code.

One might think that such an esteemed organization would prove an insurmountable challenge for a bunch of disaffected kids to hack. World-renowned, government-recognized experts against Anonymous? HBGary should be able to take their efforts in stride.

Unfortunately for HBGary, neither the characterization of Anonymous nor the assumption of competence on the security company's part are accurate, as the story of how HBGary was hacked will make clear.

Anonymous is a diverse bunch: though they tend to be younger rather than older, their age group spans decades. Some may still be in school, but many others are gainfully employed office-workers, software developers, or IT support technicians, among other things. With that diversity in age and experience comes a diversity of expertise and ability.

It's true that most of the operations performed under the Anonymous branding have been relatively unsophisticated, albeit effective: the attacks made on MasterCard and others were distributed denial-of-service attacks using a modified version of the Low Orbit Ion Cannon (LOIC) load-testing tool. The modified LOIC enables the creation of large botnets that each user *opts into*: the software can be configured to take its instructions from connections to Internet relay chat (IRC) chat servers, allowing attack organizers to remotely control hundreds of slave machines and hence control large-scale attacks that can readily knock websites offline.

According to the leaked e-mails, Aaron Barr believed that HBGary's website was itself subject to a denial-of-service attack shortly after he exposed himself to someone he believed to be a top Anonymous leader. But the person I spoke to about this denied any involvement in such an attack. Which is not to say that the attack didn't happen—simply that this person didn't know about or participate in it. In any case, the Anonymous plans were more advanced than a brute force DDoS.

Time for an injection

HBGary Federal's website, hbgaryfederal.com, was powered by a content management system (CMS). CMSes are a common component of content-driven sites; they make it easy to add and update content to the site without having to mess about with HTML and making sure everything gets linked up and so on and so forth. Rather than using an off-the-shelf CMS (of which there are many, used in the many blogs and news sites that exist on the Web), HBGary—for reasons best known to its staff—decided to commission a custom CMS system from a third-party developer.

Unfortunately for HBGary, this third-party CMS was poorly written. In fact, it had what can only be described as a pretty gaping bug in it. A standard, off-the-shelf CMS would be no panacea in this regard—security flaws crop up in all of them from time to time—but it would have the advantage of many thousands of users and regular bugfixes, resulting in a much lesser chance of extant security flaws.

The custom solution on HBGary's site, alas, appeared to lack this kind of support. And if HBGary conducted any kind of vulnerability assessment of the software—which is, after all, one of the services the company offers—then its assessment overlooked a substantial flaw.

The hbgaryfederal.com CMS was susceptible to a kind of attack called SQL injection. In common with other CMSes, the hbgaryfederal.com CMS stores its data in an SQL database, retrieving data from that database with suitable queries. Some queries are fixed—an integral part of the CMS application itself. Others, however, need parameters. For example, a query to retrieve an article from the CMS will generally need a parameter corresponding to the article ID number. These parameters are, in turn, generally passed from the Web front-end to the CMS.

SQL injection is possible when the code that deals with these parameters is faulty. Many applications join the parameters from the Web front-end with hard-coded queries, then pass the whole concatenated lot to the database. Often, they do this without verifying the validity of those parameters. This exposes the systems to SQL injection. Attackers can pass in specially crafted parameters that cause the database to execute queries of the attackers' own choosing.

The exact URL used to break into hbgaryfederal.com was `http://www.hbgaryfederal.com/pages.php?pageNav=2&page=27`. The URL has two parameters named pageNav and page, set to the values 2 and 27, respectively. One or other or both of these was handled incorrectly by the CMS, allowing the hackers to retrieve data from the database that they shouldn't have been able to get.