SECURITY (/CATEGORY/SECURITY) apple (/tag/apple/), hackers (/tag/hackers/)

Once more into the breach: How hackers compromise websites like Apple's


Marco Tabini
@mtabini

Jul 24, 2013 6:00 AM

Unless you happen to call the proverbial rock home, you've probably heard that Apple's developer websites were recently hit by a hacking attack. Though the developer site has been inaccessible since last week, the company didn't announce the intrusion (http://www.macworld.com/article/2044865/apple-explains-extended-developer-portal-outage.html) until Sunday; as of this writing, the site remains down. That outage has resulted in considerable inconvenience for app developers, not to mention the poor IT people in Cupertino who have been working around the clock to deal with the breach.

According to the company, the attackers didn't manage to get their hands on any sensitive information, but the fact remains that security hacks like this one happen with alarming frequency all over the Web, and one cannot help but wonder why websites seem to be so easy to break into.

The ABCs of computer hackery

Computers can be hacked using a variety of techniques, which typically fall into three categories: social engineering, software exploits, and hardware cracks.

Social engineering is the least technological member of this trio, although it's by no means the least sophisticated. It works by extracting access credentials from an unsuspecting user, or acquiring them from an unscrupulous operator, such as a disgruntled employee. If you've been on the Internet for more than a day, you've probably been on the receiving end of a "phishing" email, which invites you to log on to a site that looks and feels like, say, your bank's, but is in reality controlled by hackers who capture your username and password and then use it to help themselves to the money in your account.

These types of attacks, while very common against the general public, are hard to pull off against a website owner—particularly one as large and sophisticated as Apple. For one thing, IT personnel tend to be well acquainted with phishing attempts, and are usually on the lookout for them. In addition, the private nature of the internal systems that are used to manage a company's network makes them hard to spoof, unlike a banking website, which is open to the public and can be easily replicated.

The hole in the (software) donut

In practical terms, most successful attacks against websites tend to be software-based, and are often caused by the site's developers making the wrong set of assumptions. For example, a very common class of attacks called code injections (http://en.wikipedia.org/wiki/Code_injection) is caused by code that "trusts" data coming from the outside world, and thus doesn't attempt to filter out any potentially malicious commands embedded in that data. Unchecked, this kind of bug can have catastrophic repercussions, allowing a third party to gain access and even delete information stored in the site's database—such as username, emails, or passwords.

A more sophisticated vulnerability, known as cross-site request forgery (http://en.wikipedia.org/wiki/Cross-site_request_forgery), can be used to force a browser to surreptitiously navigate a target website without human intervention. If the user happens to be logged into a password-protected website, an attacker could perform unauthorized operations, including locking the real user out of his or her own account.

These are but two examples of *dozens* of possible vulnerabilities that can be caused by weak programming. And even when the website's code itself is perfect—and we know how often software is perfect—there is still plenty that can go wrong: Regardless of the operating system on which a Web server runs, it typically also makes use of hundreds of different software components that take care of everything from delivering documents to keeping the system's time. Since many of these programs have a network component, any defect in their code has the potential to become a possible entry point for a would-be hacker.

It's not all that bad

If this appears to paint a bleak picture of Web security, keep in mind that practically every potential attack can be mitigated—and almost always prevented—by using the right security measures.

In many organizations, for example, Web servers are usually kept behind firewalls (http://en.wikipedia.org/wiki/Firewall_(computing))—systems that are designed to be connected to the open Internet, and whose software is hardened in such a way as to prevent intrusion while simultaneously letting legitimate traffic through. In addition, larger companies employ sophisticated intrusion-detection software that can "sniff" network traffic and detect illicit activity before it becomes a problem; they also implement policies that that promote the development of secure software, for example by placing significant emphasis on data encryption and good programming practices.

Thus, while the occasional slip-up may still occur, it's a safe bet that, the more important a Web-based system is, the more sophisticated the level of control over its operations. What happened at Apple appears to have been a lapse in network administration (http://www.guardian.co.uk/technology/2013/jul/22/apple-developer-site-hacked) that allowed outdated software on a server to leak information about developers. While serious, this bug is unlikely to affect Apple's consumer-facing services, like iTunes or iCloud, which are probably under much greater scrutiny from the company.

There's no such thing as "safe"

This is not to say that any website is *absolutely* secure. An old dictum in the security community is that the only computer that cannot be broken into is one that is unplugged from the Internet *and* turned off.

Indeed, hardware hacks have been used to pull off some rather remarkable stunts against systems that were thought to be hack-proof because they weren't connected to any network, such as crippling Iran's nuclear program with a USB key (http://en.wikipedia.org/wiki/Stuxnet) and even reading a computer screen through a wall by detecting its radio emissions (http://en.wikipedia.org/wiki/Van_Eck_phreaking).

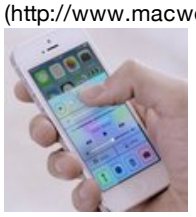
Spy stories aside, this goes to show that the potential for intrusion is a byproduct of the very functionality that makes running a website possible. While the risk can never be completely eliminated, the right combination of expertise and care, mixed with a hint of paranoia, can greatly reduce its potential impact—and nowhere is this combination of skills more likely to come into play than in a company's most important properties.

Site break-ins are serious threats that tend to get lots of publicity, but not all breaches are created equal. The kind of issue that afflicted Apple last week only involved a relatively small and obscure section of its operations, and appears to have left everyone's *really* important data, like credit card numbers, uncompromised. If anything, in fact, it's likely that what happened gave Apple's IT department a reason to tighten its internal policies, which may well result in better security for its end users in the long term.

Hands-on: Quip, the so-called

Why iOS 7's design is bold but

(http://www.macworld.com/article/2043265/why-ios-7s-design-is-bold-but-flawed.html)



Ten things I'm