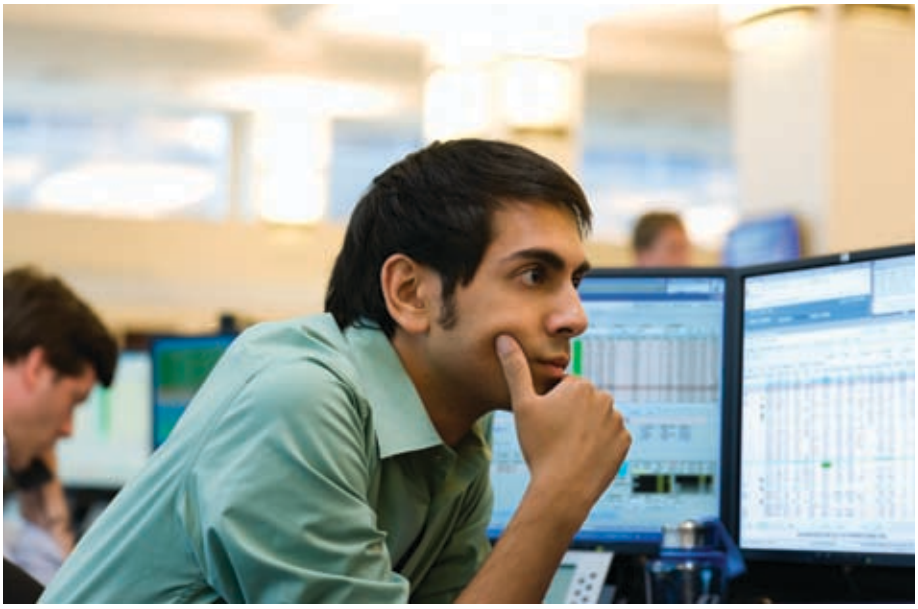

Blinded by Flash

Solving the widespread security risks Flash developers currently can't see that threaten enterprise data

The fact is that rich Internet applications (RIAs) are living up to their promise to be as powerful, flexible, and elegant as traditional, on-premise installed software. Consider the maturing of Adobe® Flash®. Flash Player is installed on roughly 98 percent of all PCs. Not long ago, Adobe Player supported a simple scripting language. And, as often is the case, languages grow more potent over time. Today, Flash is as powerful as JavaScript™.

The growth of Flash

These capabilities also add complexity—complexity that makes it easier to make programming mistakes in the applications built by developers. But Adobe Flash is a media player, not a corporate application, you say. The fact is that the risk associated with the applications built upon the Flash platform can jeopardize the security of every application on the Web server—and even the network and other domains with which it communicates. This makes Flash security a significant corporate security and regulatory issue.

Initially, Adobe Flash was popular for creating animations. However, with Adobe's release of Flex® and ActionScript® 3 (Flash Versions 9 and 10), this technology commonly has become incorporated in enterprise development and has grown well beyond animation software. In fact, Flash/Flex provides many attractive features for client-server communication, cross-domain communication, and JavaScript interaction. Owing to its large library of built-in components, Flex enables developers to write very attractive and engaging applications quickly. The platform now also incorporates many common enterprise development capabilities such as source code version control, advanced regular expression support to help developers create input validation controls, comfortable user interface design environment—as well as shifting ActionScript from a functional to an object-oriented development platform.

Blinding insecurity

To try to gain insight into the level of security that Flash developers were incorporating into their applications, we conducted some research. We searched the Internet for SWF programs that included admin and login strings. After analyzing 150 of these applications, we found that 23 had the access information—including cleartext passwords—actually hardcoded within the application.

That's certainly not a good start and, unfortunately, it's only the beginning of the issues we found. For instance, because Flash supports cross-domain access, it's possible for attackers to increase their access privileges on other referenced networks. Because the vulnerability easily is avoidable, while the potential negative impact is so great, let's focus on Flash-related cross-domain issues for a moment.

Attacks on SWF applications grow.

- **July 2006,** *MySpace ad injects malware into 1.07 million computers*
- **February 2007,** *Malware found in Windows Live Messenger ads*
- **September 2007,** *Yahoo feeds Trojan-laced ads to MySpace and PhotoBucket users* (also affected: TheSun.co.uk, Bebo.com, and UltimateGuitar.com)
- **November 2007,** *Whitepages online and Bigpond ads 'hijack' users*
- **December 2007,** *Hackers Use Banner Ads on Major Sites to Hijack Your PC* (The Economist, MLB.com, Canada.com, etc). *Redirect function encrypted; Malware bandits go looking for goals on ESPN's Soccernet.com*
- **January 2008,** *Rogue ads infiltrate Expedia and Rhapsody; ITV Web Site Forces Scareware Package Through Banner Ads*
- **April 2008,** *Yahoo! pimping malware from banner ads; Fake FedEx Advertisement*
- **August 2008,** *Malvertisement epidemic visits Newsweek.com*

Just as is the case with JavaScript, Flash Players natively are constrained from issuing asynchronous HTTP requests to foreign domains. In this way, SWF files can't access data on other domains. While this improves security, it can reduce flexibility in some cases. For instance, a company may need to access movies or data on another domain that it owns, or entirely different companies may want to exchange information for mashups. For these cases, Adobe added the use of cross-domain policy files. These files are simple XML statements that provide the Flash Player the ability to access data from a certain domain—without any security dialogs or any access prompt. The problem is: Policy files can contain wildcards and developers seem to like to use them. Obviously, if not implemented properly, cross-domain XML policies can make it all too easy for attackers to increase their access privileges on other referenced networks—a serious security problem, without question.

Jeremiah Grossman, chief technology officer at WhiteHat Security, had similar questions. After evaluating various Fortune 500 sites as well as the Alexa 100, Grossman found that 8 percent of the Fortune 500 had crossdomain.xml policy files and 2 percent had policy files that had wildcards that enabled access to virtually any domain. When it came to the search of the Alexa 100, Grossman found that 36 percent of sites contain the crossdomain.xml policy file, and the number of such highly insecure crossdomain.xml implementations tripled to 6 percent.

Hardcoded passwords and ill-configured cross-domain policy files are only the beginning. If applications built on the Adobe Flash platform aren't designed properly, it's possible for attackers to inject code of their choice into the client applications. Also, they're commonly vulnerable to spoofing, browser script injection, data injection, DNS attacks, Internet worms, and many other security exploits.

Despite the risks, there's no reason why enterprises can't embrace Flash. But they need to ensure that these applications are developed securely. Adobe has published a number of best practices that, if followed, will improve significantly the security of applications built with the Flash platform. These practices are quite the same as the proper practices associated with any type of software development: validate inputs, don't store sensitive information within the application itself, make sure all communication across domains is secured, and encrypt data in transit through SSL, among others. Another good starting point is the resources made available by the Open Web Application Security Project (OWASP), which is dedicated to Web application security and education.

HP Web Security Research Group introduces SWFScan

To help organizations better secure applications developed using the Adobe Flash platform, the HP Web Security Research Group developed SWFScan (pronounced Swiff Scan). SWFScan, currently available for general download, identifies and provides remediative information for many of the vulnerabilities that affect SWF applications. SWFScan works like many application scanners: simply point SWFScan at a URL that contains a Flash file, or load a local Flash file.

Creating more secure SWF Web applications

Flash development needs the same care and due diligence as any Web application: Don't store passwords, use encryption when necessary, be careful with cross-domain polices, and always follow good coding practices. Adobe's Peleus Uhley has assembled a very thorough security checklist for developing SWF applications.

Data validation

- Check for malicious URLs in all imported data.
- Escape special characters when writing into an HTML text field.
- Validate all data received from outside resources such as LocalConnections, LoadVariables, and XMLSockets.
- Use encrypted protocols where possible to prevent man-in-the-middle attacks.
- Check for returned exceptions as they may be an indication of malicious data.
- Be aware of situations in which remotely loaded SWF files can script into your SWF and alter your variables and functions.
- Utilize ActionScript 3.0 rather than ActionScript 2.0 to prevent uninitialized variable attacks.
- Protect sensitive data.
- Set the secure flag on local shared objects when using SWF files with SSL.
- Do not override the default path on the local shared object or, if you must, set the most restrictive path.
- Do not set the secure flag to false within crossdomain.xml files.
- Do not use the allowInsecureDomain() method for SWF communication.
- Limit allowDomain() settings to specific domains.
- If SSL is used, use it consistently throughout the entire application workflow.
- Do not override the exactSettings setting in Flash Player.
- Leverage third-party libraries for additional cryptography.

Cross-domain communication

- Use specific domains in crossdomain.xml files.
- Leverage meta-policies to limit cross-domain and socket policies on your server.
- Do not use wildcards in domain or header settings within the cross-domain or socket policy file.
- Consider the most appropriate form of communication for the task at hand (i.e., a single method through LocalConnection versus cross-scripting).
- Use full domain names within allowDomain() settings.
- Be aware of the issues with import loading an SWF file into the current security domain via Loader.loadBytes or setting the securityDomain in LoaderContext.
- Do not override the exactSettings flag in Flash Player.
- Do not use unvalidated FlashVars as arguments to loadPolicyFile().

Preventing cross-site scripting attacks

- Set appropriate allowScriptAccess and allowNetworking parameters within the HTML code.
- Perform data validation on variables sent to URL functions to ensure only http:// and https:// protocols are allowed; validate that the URL is for an allowed domain or use relative URLs.
- Escape special characters placed within HTML text fields.
- Do not use HTML text fields unless HTML support is needed.
- Compile the SWF for more recent Flash Player versions.
- Encourage users to have the latest version of Flash Player to view your content.

Spoofing

- Set masks on loaders for externally loaded SWF files.
- Only allow trusted SWF files to use full-screen mode.

Information disclosures

- Do not store sensitive encryption keys or passwords in a SWF file because it can be decompiled.
- Using the compiler flags such as "Protect From Import," "Omit Trace Actions," and "Permit Debugging" will not prevent determined attacks.
- Be aware that data stored in shared objects also are readable by third parties that have file system access since the data within the shared object is not encrypted.
- Use available cryptographic libraries when possible for encrypting data.
- Use an SSL connection consistently for more secure Internet communications.

SWFScan then will decompile the SWF byte code and generate the ActionScript source code. SWFScan then performs static analysis on the ActionScript code to gain an understanding of the behavior of the application. It then vets the code for nearly three dozen known security issues. Finally, a report on all identified vulnerabilities is generated, including:

- Source code causing vulnerabilities
- Implications of the specific vulnerabilities found
- Best practice guideline to help with remediation (These guidelines are based on a collaboration among Adobe and many industry experts.)
- Additional information that is useful for the manual inspection of the SWF files, such as networking calls, external domain requests, and more

Developers can leverage SWFScan to not only audit the security of their own applications, but also third-party SWF applications, including banner ads and other objects—before embedding them into their Web sites. This capability is vital for situations where the application source code is not available: SWFScan will generate the source code necessary for thorough evaluation.

This ability to generate and analyze the source code is critical. Properly addressing Flash security is complex—in many ways much more complex than tradition Web applications. The HP Security Research Group has invested considerable resources and expertise to gain an advanced understanding of the inherent vulnerabilities that can occur when developing applications with the Adobe Flash platform. We've found that in order to build properly secured applications, as well as vet third-party applications, surface scans are not enough. You need to analyze the ActionScript itself.

For instance, while SWF applications can be exploited from surface attacks, they can also be exploited with sensitive information disclosure, or incorrect cross-movie scripting permissions and cross-site scripting attacks. The vulnerabilities that make these attacks possible lay within the ActionScript, and are not obvious or easily detectable through surface analysis alone.

Conclusion

During the past two decades, many developers have been repeating the same mistakes—failing to validate inputs; inserting information that can be used to launch attacks within the application code; and failing to manage privileges properly, among many others. These mistakes were made during the stand-alone applications in the 1980s, the distributing computing architectures during the 1990s, and the initial wave of Web applications earlier this decade. Let's not continue the cycle as these new development platforms gain in popularity and functionality. The security of your organization, and all of our data, will depend on it.

As you've seen, it's not impossible, nor even very difficult, to develop secure SWF applications. Developers need to apply the best practices outlined by Adobe—and continuously check SWF applications as well as all installed applications, Web applications, and Web servers for programming and configuration errors that lead to security breaches and systems that are not in compliance with many state, federal, and industry mandates.

While criminal hackers have altered their tactics to target Web servers, Web sites, and applications, it's as important as ever that enterprises make certain that they're investing in the right tools to win the fight against the latest threats. And with the continued growth of Web applications, Web-based attacks, and data leaks will remain the most popular vector of attack: Flash applications will be no different. Fortunately, through the HP Application Security Center, enterprises can secure their applications even with limited investment. To learn more about the HP Application Security Center, and how HP can help your organization develop a more secure and sustainable infrastructure, please contact us at (866) 774-2700 x1, **qmsecuritysales@hp.com** or visit our Web site at **www.hp.com/go/securitysoftware**.

Technology for better business outcomes

To learn more, visit www.hp.com

4AA2-4752ENW, March 2009