

AUGUST 2004

Macromedia Flex

Macromedia Flex is designed to facilitate the deployment of Rich Internet Applications, which combines desktop software functionality with the broad reach and low cost deployment of the web. The premise behind the Flex solution is to establish a presentation layer independent from the application server logic to provide a robust, feature-rich, and portable client-side execution environment. While conducting a product penetration assessment against the Flex presentation server, @stake found that Flex provides a platform for developers that meets application security best practices to defend against client-side attacks.

Overview

@stake's product evaluation provided a practical demonstration of the Macromedia Flex Presentation Server security controls by attempting to circumvent those features designed to protect the confidentiality and integrity of the application, data, and client operating environment. In this document, we review some of the security controls that are present in Flex. As with any development system, Flex developers will receive the most benefit from the security of the Flex Presentation server by designing and developing their applications using these features.

Flex Platform Overview

The Flex platform combines client and server technologies that allow developers to create and deploy highly-interactive web applications. Flex introduces MXML, an XML-based language designed to enable dynamic generation of SWF (Flash) files. MXML is a declarative language for describing user interface controls and managing data accessed via web services, remote procedure calls to Java objects, and XML over HTTP. Flex also uses ActionScript 2.0, based on ECMA-262 Edition 4, in conjunction with MXML for client-side business logic.

The Flex server compiles MXML and ActionScript source code to generate a Flash application that is then delivered to the client along with any required external data. Once on the client, the Flex application executes within the Flash Player.

Flex Server Architecture

Macromedia's Flex presentation server architecture is a native Java (J2EE) application that runs on leading Java application servers, including IBM WebSphere, BEA WebLogic, Macromedia JRun, and Apache Tomcat. A native .NET version is in development.

The application server can be configured securely using standard configurations such as those listed on the following websites:

http://java.sun.com/blueprints/guidelines/designing_enterprise_applications/security/

http://www.atstake.com/research/reports/eval_ms_ibm/

<http://edocs.bea.com/wls/docs81/security/>

The Flex server benefits from all of the intrinsic security benefits inherently available to Java-based applications. Flex presentation server makes use of type safety to prevent programmatic errors, input validation, a robust security model, an abundance of managed code, inherent buffer overflow protection, and ubiquitous and well-documented security standards.

Flash Player

The Flash Player provides a security sandbox to isolate applications and the host operating system. The Flash Player sandbox security has a significant advantage over traditional web-enabled application components, such as ActiveX, that require users to trust code that often has complete access to the operating system environment.

The default configuration of the Flash Player's sandbox provides security by segmenting execution privileges between the Flex application and the operating system. Similar to Java's sandbox technology, the Flash Player's sandbox prevents unauthorized access to the operating system environment as well as other local instances of the Flash player.

Data Access White Lists

Flex uses a centralized "white list" for administratively controlling access to all data. Data and object access is specified in the white list through the use of wild cards and URLs. Data access requests going through the proxy servlet are normalized to prevent character decoding and interpretation attacks, and then compared against Flex's internal white list. Because this approach controls both user input and information returning from the backend using a sophisticated permissions model, it is more powerful than many other access control systems.

As part of the security evaluation, @stake attempted to subvert the white lists via resource starvation, input validation failures, and circumvention of URL encoding within the Flex server. Flex demonstrated resiliency from these attacks and confirmed the server's ability to appropriately handle the most common Internet attacks. This robust input validation proved to be powerful in diminishing the ability of a malicious attacker to obtain confidential information or disrupt Flex application services.

Industry Standards

Flex supports many underlying web technologies, which are extended to provide new functionality. The use of standard web building blocks allows Flex to benefit from common security technologies.

Because existing XML and SOAP standards are used for transport, HTTPS is supported for all operations. For all applications handling sensitive data, @stake recommends that message transport and communication be encrypted and signed to protect the confidentiality and integrity of an application's data. Industry standard SSL-protected protocols are available through Flex. Interaction supported by the underlying web server allows for seamless client/server identification and encryption. By configuring Flex resources to use SSL, confidential information will be protected from man-in-the-middle attacks regardless of the end users' environment.

As a J2EE application, Flex supports authentication and authorization techniques that are native to the J2EE environment. Flex is capable of interacting with role-based and declarative security at the application or EJB level and supports industry standard J2EE security methods. It should be noted however that, as with all Java applications, protection is dependent on the developer properly designing and implementing J2EE security best practices. Flex users need to be cognizant of security best practices for J2EE development. The Macromedia Flex installation guidelines describe J2EE standards that should be followed during deployment. Additionally, Sun Microsystems provides its own set of J2EE security guidelines at http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/Security.html.

Flex Tags

Flex provides an mx:HTTPService tag designed to maintain information accessed via HTTP Get and Post methods. This tag is processed by the server during compilation and therefore can not be modified by the client-side.

Session Management

Flex leverages the underlying application server's ability to track and provide information on user sessions. During testing, @stake attempted to steal a Flex-initiated session from one user and instantiate the user's session on another host. Attempts to steal and reuse user sessions were unsuccessful because Flex's server uses the application server's session management API that prohibits attacks that can lead to account compromise or the insertion, modification, and deletion of data exposed by the session.

Identity Management

Identity management is useful for providing strong user authentication and fine-grained access control. As there are no specific hooks to identity management systems within the Flex presentation server, underlying identity management logic is performed at the application server, where the Flex presentation server integrates with the application server to take advantage of attributes provided by the identity management system. During our test of the Flex presentation server, we did not identify any issues with Flex and the integration of identity management services with the application server.

Conclusion

Based on @stake's evaluation of the Flex presentation server, it appears that Macromedia has developed a strong information protection model against client-side threats. Flex's architecture mitigates many common client-side attacks such as cross-site scripting, denial-of-service, SQL injection, man-in-the-middle and session hijacking. Server-side security is maintained by leveraging J2EE security by mitigating common attacks against infrastructure components, such as buffer overflows, heap corruption, and cross-site scripting.

Nevertheless, the security features built into the product still rely on proper configuration and role separation by developers deploying this technology. If white list security is correctly managed, server security patches are applied in a timely manner, and best practices in regards to EJB security are followed, the benefits of the Flex security model are significant. The Macromedia Flex installation documentation identifies the J2EE security standards that should be followed during deployment.

The dependence upon the security knowledge and sophistication of developers is not a security issue unique to the Macromedia Flex Presentation Server. It is an industry-wide problem that needs to be addressed by improving the application security principles within the J2EE developer community.

@stake commends Macromedia's proactive stance towards security and strongly recommends users to read, understand, and adhere to Macromedia's guidelines. @stake also recommends diligence from Macromedia and the Macromedia user community in updating and monitoring best practices as new features and tools are added to Flex.

To learn more about Macromedia Flex, visit:
<http://www.macromedia.com/software/Flex>

About @stake, Inc.

@stake, Inc., the premier digital security consulting company, provides security services and award-winning products to assess and manage risk in complex enterprise environments. The company's SmartRisk services cover key aspects of security, including applications, critical infrastructure, wireless and wired networks, storage systems, education, and incident readiness. @stake consultants combine technical expertise with a business focus to create comprehensive security solutions to mitigate risks and maximize results. @stake clients include six of the world's top ten financial institutions, four of the world's top ten independent software companies and seven of the world's top ten telecommunications carriers.

As the first company to develop an empirical model that measures Return On Security Investment (ROSI), @stake keeps security investments in line with business requirements. Headquartered in Cambridge, MA, @stake has offices in London, Chicago, New York, Raleigh, San Francisco, and Seattle. For more information, go to www.atstake.com.

Reproduction guidelines: you may make copies of this document unless otherwise noted. If you quote or reference this document, you must appropriately attribute the contents and authorship to @stake. Opinions presented in this document reflect judgment at the time of publication and are subject to change. While every precaution has been taken in the preparation of this document, @stake assumes no responsibility for errors, omissions, or damages resulting from the use of the information herein. Products or corporate names may be trademarks or registered trademarks of other companies and are used only for the explanation and to the owner's benefit, without intent to infringe.