



September 2003

*Building Secure Internet Applications
With ColdFusion MX 6.1*

© 2003 Macromedia Inc. All rights reserved.

The information contained in this document represents the current view of Macromedia on the issues discussed as of the date of publication. Because Macromedia must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Macromedia, and Macromedia cannot guarantee the accuracy of any information presented after the date of publication.

This Guide is for informational purposes only. **MACROMEDIA MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.**

Copyright 2003 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo, ColdFusion, the ColdFusion logo, Dreamweaver, UltraDev, Flash, JRun, HomeSite, Macromedia Generator, and Macromedia Spectra are trademarks or registered trademarks of Macromedia, Inc. Other brands may be trademarks or registered trademarks of others.

Macromedia, Inc • 275 Grove Street • Newton • MA • 02466

www.macromedia.com • info@macromedia.com • (617) 219-2000

Table of Contents

Executive Summary.....	4
Introduction.....	5
Security Exposures.....	5
Snooping or Eavesdropping.....	5
User Impersonation.....	5
Unauthorized Access.....	6
Security Solutions.....	6
Data Encryption.....	6
User Authentication.....	7
Access Control.....	8
ColdFusion MX Security.....	9
Overview.....	9
Development Security.....	9
The ColdFusion Administrator.....	10
Remote Development Services.....	10
Page Encoding.....	11
Runtime Security.....	11
Data Encryption in ColdFusion Applications.....	11
User Authentication in ColdFusion Applications.....	12
User Access Control in ColdFusion Applications.....	14
Application Access Control in ColdFusion.....	15
Multiple Server Instances.....	16
Conclusion.....	17
Appendix.....	17
Disabling RDS access.....	17
Additional ColdFusion Security Resources.....	18

Executive Summary

This document explains how ColdFusion MX 6.1 provides powerful, yet easy to implement data encryption, user authentication, and access control features that address the most common web application security risks. It defines some of the security concerns on the web today and illustrates the features of ColdFusion that help answer questions such as:

- How sensitive application data can be protected while in-transit between client and server machines
- How to secure the remote development of web applications
- How to validate a user's identity and credentials against an authorization store
- How to grant or deny specific application functionality based on a user's security clearance

Introduction

Web application developers are very concerned about security, and rightfully so. The nature of the web — global access, ease of connectivity and interaction, and lack of true client control — creates an environment where application misuse or abuse can flourish. As such, almost any discussion of web applications and data integration quickly becomes a discussion of security. Web application developers must fully understand the security risks in order to address the legitimate concerns, while ignoring the tabloid-style hype.

Security Exposures

Web application security risks fall into three major categories:

- Snooping and Eavesdropping
- User impersonation
- Unauthorized access

Snooping or Eavesdropping

The risk of having someone “overhear” data being sent over the Web is a primary concern when sending confidential data, such as credit-card information, over public connections.

On routed IP networks like the Internet, it is relatively difficult to eavesdrop on specific connections without having privileged access to the local ISP routers. Communications packets of any given message could be sent over completely different routes to get from the sender to the receiver, rendering the coherent snooping of the transmission nearly impossible.

The risk of “packet sniffing” is still there, however, especially in Local Area Network (LAN) settings.

User Impersonation

Without proper authentication control, the risk of non-trusted users gaining access to secure information by impersonating trusted users is a very real risk. User authentication is the foundation of Internet based application security, and inadequate authentication leaves applications vulnerable to attack.

Unauthorized Access

The risk of exposing sensitive information to unauthorized users is the biggest and most complex security risk, because the Internet effectively links every computer to one large network. While completely allowing or disallowing access to a given system or data source remains relatively straightforward, partial access remains risky. At the same time, access to distributed systems is what makes the Internet a valuable business tool.

For example, it is easy for a large bank to put up a public, freely accessible site where NO individual account information is available, but much harder to create an account maintenance site where users have exclusive access to their personal accounts.

Security Solutions

Every Internet based application needs to address these basic security concerns. Over the past few years, several technologies and techniques have evolved as standard mechanisms with which to secure web applications.

ColdFusion integrates with these technologies to provide a seamless, unobtrusive security framework. The risk categories identified earlier:

- Snooping and Eavesdropping
- User Impersonation
- Unauthorized Access

are handled by the following security mechanisms respectively:

- Data Encryption
- User Authentication
- Access Control

An understanding of the basic standard solutions will later clarify the means by which ColdFusion integrates with them to provide a complete security framework.

Data Encryption

HTTP data is often transmitted over open lines, shared data channels, and public access providers. If electronic snoops were to “eavesdrop” on this connection they would be able to copy every byte of data transmitted. While not a common occurrence, this kind of theft is technically possible, making this a legitimate concern, particularly for sensitive data such as credit-card information.

Data encryption is a mechanism used to prevent data from being stolen in-transit between the client and server machines. Electronic thieves can still grab the data being transmitted, but they will find it useless in its encrypted state.

The most common form of data encryption used by web-based applications is Secure Sockets Layer (SSL). SSL is a security protocol that provides Internet application protocols (like HTTP) with data encryption using public key cryptography.

Most web servers support SSL, allowing administrators to install a private key that is used to decrypt inbound data and encrypt outbound data. Once installed, the web server automatically encrypts or decrypts data as it is received or transmitted.

User Authentication

User authentication is the process of making sure the user is who he claims to be. User authentication is used by all network operating systems (NOS), e-mail packages, and even devices such as ATM machines. Performing user authentication involves prompting the user for a unique identification along with some form of verification – information that no one other than the user could know. Passwords and PIN numbers are common verification mechanisms. This form of authentication is often referred to as “challenge and response.” The application challenges the user to prove his or her identity, and allows access if a valid response is returned.

Some of the more popular challenge response methods include:

- Password Authentication
- Digital Certificates
- Electronic Security Cards

Password authentication is by far the most common, but digital certificates and electronic security cards are rising in popularity.

Password Authentication

In password authentication, the application prompts the user for a user ID and password, which is verified against an account database, and access is either granted or denied based on that verification. While this is the most popular form of authentication, it is also the least secure. The biggest hole in this form of security is the password itself. Recent statistics indicate that most networks do not require users to change passwords frequently, do not prevent password reuse, and do not enforce an adequate minimum password length. Most of the recent high visibility “hacks” into corporate or government networks resulted from poor password protection, allowing electronic thieves the ability to guess passwords and gain access.

Additional problems with password-based authentication include forgotten passwords, and unattended, authenticated workstations. In the first case, the authorized user cannot be unauthenticated while in the latter, the authorized user authenticates himself and then leaves his authenticated workstation available to be accessed by unauthorized users.

Digital Certificates

To solve some of the problems with passwords, many sites are now taking advantage of “digital client authentication”. These certificates are obtained from a trusted certificate authority, and installed into the browser. The certificate contains signature and key information that can be used to validate its authenticity. This form of authentication provides web servers the ability to query the client browser for a digital certificate, which uniquely and safely identifies a specific browser. Users need not remember passwords; in fact, there are no passwords to remember. The web server and client exchange certificates, and access is granted or denied accordingly. The biggest problem with digital certificates is that they identify clients, not users. They are therefore best suited for users who do not use multiple computers on a regular basis.

Electronic Security Cards

Another popular technology is the use of electronic smart cards or security tokens. There are several variants of these technologies on the market, but they all basically act as electronic identification cards. These cards require special hardware or software to be installed on the network, and all authorized users are given a card that uniquely identifies them. Some cards are swiped through magnetic readers (similar to credit cards or ATM cards); others display digital IDs that are typed into a computer. These cards can be highly secure; unless lost or stolen, in which case they could be misused.

The common denominator here is that once a user correctly responds to a provided challenge, authentication is complete, and it is safe to assume that the user is who he or she claims to be.

Access Control

Application security is rarely implemented as simply as “deny all access” or “grant all access.” Users are usually granted access to particular features or components based on security clearance, group affiliation, or some other criteria. The process of determining which features and options are visible to specific users is known as “access control.”

Access control typically is used to accomplish one or more of the following:

- Restricting access to an entire application
- Restricting access to specific functions within an application

- Restricting access to specific features within application pages
- Dynamically changing available options to make them user specific
- Restricting access to specific data within an application
- Enforcing levels of access for specific data: no access, read-only, read-write, etc.

As previously explained, web server-based user authentication is used to identify users, and to grant or deny access to entire applications. While this may provide basic security control, it does not provide developers with application level access control. However, because web server user authentication is an ideal method of identifying users, application level access control is often built on top of this.

Unlike data encryption and user authentication, access control is very application specific. As such, application developers must implement this level of security themselves, inside the application pages. ColdFusion provides special tags and functions to make programmatic access control much easier to implement, as you will see in the following sections.

ColdFusion MX Security

Overview

ColdFusion MX 6.1 addresses each of the three aforementioned security concerns during both of the key phases of an application: development and deployment. Development security addresses the safe use of ColdFusion in an application's development process – securing access to the ColdFusion Administrator, and securing remote development carried out with Macromedia Dreamweaver MX or Macromedia HomeSite+. Runtime security answers security concerns for deployed applications – protecting application data, authenticating users of the application and granting or denying access to certain functionality based on the roles assigned to those users. Runtime security can also be used to control which resources the ColdFusion application itself may access, such as data sources, file system resources and so forth.

Development Security

When developing ColdFusion applications, two areas are typically used; the ColdFusion Administrator, and remote development serviced (RDS) with the integrated development environment (IDE). Both Dreamweaver MX and HomeSite+ support RDS..

The ColdFusion Administrator

The ColdFusion MX Administrator is used to configure and maintain the ColdFusion application server. It is a web-based application that can be accessed using any web browser, from any computer with an Internet connection. It is used to manage configuration options such as ColdFusion data sources, debugging settings, global server settings, and application security configurations.

The Administrator installs with secure access enabled. A password must be created during product installation, but can be changed at any time by an authorized user. Each time the Administrator is accessed, this password must be supplied. Once successfully authenticated under this password, users have complete access to all Administrator services supported.

Since the ColdFusion Administrator is itself a web-based ColdFusion application, normal web server security may also be enabled in addition to the native ColdFusion password security. For example, the web server could configure the ColdFusion Administrator site to be protected by SSL, as was discussed previously, thereby adding another layer of access security.

In some cases, such as those hosting applications for multiple departments or clients on the same server, it may be necessary to provide Administrator access to multiple individuals. To prevent the potential security risk in this case of one Administrator user inadvertently changing a setting that would impact another application, ColdFusion MX Enterprise 6.1 can be deployed in multiple server instances, providing each site or application with its own ColdFusion Administrator. The owner of each ColdFusion instance can safely make changes within the Administrator that will not affect other instances. More information on the security benefits of multiple server instances will be discussed later in this document.

Remote Development Services

ColdFusion has included Remote Development Services (RDS) since version 3.1. RDS allows developers using Macromedia Dreamweaver MX, HomeSite+, or older versions of ColdFusion Studio to develop, deploy, and maintain remote ColdFusion applications on remote ColdFusion servers. With RDS, the tools interact with the remote file system and data sources by connecting to the RDS on the target ColdFusion server.

Securing RDS Development

RDS authentication is controlled by the ColdFusion server. A single password secures the entire server under RDS access, allowing full access to authenticated users. Password-restricted access to server resources for RDS is enabled or disabled from the ColdFusion Administrator.

During development, SSL encryption may be also used as an option to protect all RDS communications between Dreamweaver MX and the ColdFusion server. This includes remote access to server data sources and drives, provided that both are accessed via RDS.

It is very important to disable RDS access in production environments. In earlier versions of ColdFusion, RDS ran as a separate service or process and could be disabled by disabling the service. In ColdFusion MX, RDS is seamlessly integrated into the main service or process. To disable it, you must disable the RDServlet mapping, as described later in the Appendix.

Page Encoding

ColdFusion MX also includes a utility called cfencode, which obscures the source of ColdFusion pages that comprise an application. Although this technique cannot prevent determined hackers from reading the contents of a CFML page, it does prevent inspection of the pages. Moreover, since ColdFusion pages are processed on the server, a properly configured web server should prevent users from being able to view the source code at all, whether encoded or not.

Runtime Security

While development security is necessary, security of deployed applications is essential, especially since many applications built in ColdFusion are accessible from the public Internet. Within each application built and deployed on ColdFusion, consideration must be given to the three major security concerns: encryption, authentication and access control. ColdFusion MX includes product features and language elements that allow developers to easily address each of these.

Data Encryption in ColdFusion Applications

Page Transmission

As with any data sent over public data lines, information sent to and from ColdFusion applications from clients' Web browsers can be "listened to" unless the communication is encrypted. This is of particular concern when confidential information (such as credit card numbers) is transmitted. While ColdFusion does not itself provide encryption between the client browser and server, it rides transparently upon standard web server encryption. Encryption technologies such as Secured Sockets Layer (SSL) and Virtual Private Networking (VPN) have matured to be extremely secure, and incur minimal performance overhead.

SSL Support

ColdFusion also supports SSL directly in its Internet protocol tags. The Sun JSSE library, which supports 128-bit encryption, is used to support SSL in HTTP operations, and ColdFusion provides V2 SSL encryption in LDAP operations.

Lastly, ColdFusion MX also includes a built-in web server for developing, testing, and debugging ColdFusion applications. This server supports SSL to encrypt and decrypt page requests and the pages returned by the web server.

String Encryption

The CFML language includes an encryption function, `Encrypt`, which provides symmetric key-based 32-bit encryption. It uses an XOR-based algorithm that uses a pseudo-random 32-bit key, based on a seed passed by the user as a function parameter.

Although the `Encrypt` function provided by ColdFusion offers some degree of security, there are other public algorithms, such as DES (Data Encryption Standard), a U.S. and international standard, which are stronger. One measure of the strength of an algorithm is how difficult it is for a hacker to figure out the key used to encrypt a string. The key used by the ColdFusion algorithm is 32 bits long; however, there are other algorithms that use keys up to 1028 bits long. If an application requires this type of more sophisticated algorithm, ColdFusion allows you to easily integrate with a third party encryption library.

An additional useful encryption function is `Hash`. The `Hash` function takes a variable-length string and converts it into a 32-byte, hexadecimal string, using the MD5 algorithm developed by Ron Rivest. The one-way hash function is important in securing information and in providing some measure of data integrity. A one-way function is one that provides a conversion, however, there is no means of reversing the conversion. In particular, the MD5 hash algorithm used in ColdFusion MX converts a variable-length string to a fixed-length string. The resulting string is always the same length, 32 bytes. Although you cannot use a hash function to determine whether two strings are equal, you can use it to get a reasonable assurance of accuracy.

A hash function can be applied to a password, thus creating a "fingerprint" of the password that can be stored in a database. Because no one can reverse the conversion, storing a password in this fashion is very secure. Likewise, the hash function can be applied to a document that is to be stored in a content management system. Whenever a new document is added, the content management system compares the hash values of existing documents with the hash value of the new document to ensure that it is not already in the system without having to compare the document with the complete contents of every other document.

User Authentication in ColdFusion Applications

As was mentioned earlier, "challenge and response" authentication is a common way of verifying the identity of a user. The application challenges the user to prove his or her identity, and allows access if a valid response is returned. ColdFusion applications can be secured using either, or both, of the following forms of challenge and response authentication:

- Application authentication, where the ColdFusion application authenticates the user and does not allow access to the application by users without valid login IDs
- Web server authentication, where the web server authenticates the user and does not allow access to the website by users without valid login IDs

Application Authentication

ColdFusion can control the entire process of authenticating a user. An application displays a login page, checks the user's identity and login against its own authorization store, such as an LDAP directory or database, and logs the user into ColdFusion using a CFML tag. The application can then use security functions to check the user's roles or identity for authorization before running a ColdFusion page or specific code on a page. This will be discussed fully in the access control section below.

In ColdFusion 5 and earlier, the application developer would write all of the code necessary to manage whether or not a user is logged in, validate a user's identity and credentials against an authorization store, and so forth. In ColdFusion MX, a framework of tags and functions are provided to handle much of the authentication tasks automatically.

Web Server Authentication

All major web servers support basic HTTP authentication. Some web servers also support other authentication methods, including Digest HTTP authentication and Microsoft NTLM authentication. Bear in mind that basic HTTP authentication sends the user name and password in a base64-encoded string with each request. If you do not use SSL (Secure Sockets Layer) for all page transactions, the user ID and password are not protected from unauthorized access.

In web server authentication, the web server requires the user to log in to access pages in a particular directory, as follows:

1. When the user first requests a page in the secured directory, the web server presents the user with a login page.
2. The user fills in the login page and submits it.
3. The web server checks the user's login ID and password, using its own user authentication mechanism.
4. If the user logs in successfully, the browser caches the authentication information and sends it in an HTTP Authorization header with every subsequent page request from the user.
5. The web server processes the requested page and all future page requests from the browser that contain the HTTP Authorization header, if it is valid for the requested page.

Web server authentication can be used independently of any ColdFusion security features. In this case, you configure and manage all user security through the web server's interfaces. However, you can also use web server authentication in concert with ColdFusion application authentication, which would then allow the use of ColdFusion security for user access control (which will be discussed below). In such a case, the application can rely on the web server to authenticate the user against its user and password information, and does not have to display a login page. The developer then implements the ColdFusion security framework tags to log the user into the ColdFusion user security system.

User Access Control in ColdFusion Applications

After authenticating a user's identity with one of the methods discussed above, it may be necessary to grant certain users access to some functionality and deny the same functionality to others. As was mentioned earlier, this process of determining which features and options are visible to specific users is known as "access control." For example, in an employee database, all users may be members of either an employee group or a contractor group. They may also be members of roles that identify their department, position in the corporate hierarchy, or job description. For example, someone could be a member of some or all of the following roles:

- Employees
- Human Resources
- Benefits
- Managers

In such a company's intranet, the Human Resources department may maintain a page on which all employees can access timely information about the company, such as the latest company policies, upcoming events, and job postings. All employees should be able to read the information, but only certain authorized Human Resources employees should be able to add, update, or delete information.

Roles enable application resources to be controlled without requiring the application to maintain knowledge about individual users.

ColdFusion MX allows developers to programmatically bind application functionality to those users identified by their roles in two ways; CFML language functions and component roles.

CFML Roles Implementation

The CFML language includes a function called `IsUserInRole` to programmatically control application functionality based on a user's role. For example, the following code evaluates whether or not a user is authenticated as an application administrator or user and directs the user to a specific page based on that identity:

```
<cfif IsUserInRole("Admin") >
    <!-- Authenticated user is an administrator -->
    <cflocation url=" http://localhost/admin_menu.cfm" >
<cfelse IsUserInRole("User") >
    <!-- Authenticated user is a user -->
    <cflocation url=" http://localhost/user_menu.cfm" >
</cfif>
```

ColdFusion Component Roles Implementation

ColdFusion Components (CFCs) allow developers to bind the execution of a component's methods to an authenticated user's role(s). An attribute of the `cffunction` tag takes a role or list of roles as a parameter and then only allows those users authenticated as a member of that role to execute the function. If a user who is not a member of the specified role(s) attempts to execute the component method, an error is thrown and the attempt is aborted.

This technique also allows rich internet applications built with Macromedia Flash MX to be integrated with ColdFusion role-based security. Using the Flash Remoting service, which is included in all editions of ColdFusion MX, Flash applications can pass username and password information to ColdFusion MX using the `setCredentials` function in ActionScript. These credentials are then used to authenticate the user, as described above, in the same way that authentication is performed for any ColdFusion page. Thereafter, each Flash Remoting call is checked and the user must be assigned to the correct role to access the component function. Otherwise, an error is thrown and the attempt is aborted.

Application Access Control in ColdFusion

While the previous section discussed granting or denying content or functionality to a user based on his or her identity, application access control involves granting or denying functionality to a page based on its location in the file system. ColdFusion MX Enterprise 6.1 accomplishes this with a feature called Sandbox Security (called Resource security in the Standard Edition), which uses the location of ColdFusion pages to control access to ColdFusion resources.

This feature is quite important considering the power of many CFML tags and functions. While they allow the development and deployment of quite advanced applications, certain features, if misused, could cause significant problems. They could allow applications to modify the Windows registry, read, write, and delete files on the file system, execute a process on the server and more.

Additionally, many servers host several sites or application on the same machine. In such cases, developers of one site should only have access to their own files, data sources, and other resources.

With sandbox security, a set of resource limitations (or a “sandbox”) is defined and named in the ColdFusion Administrator. This sandbox identifies which tags, functions, and resources (such as files, directories, and data sources) can be used by ColdFusion pages located in and below a designated file system directory. In Standard edition, a single sandbox can be configured and it applies to all ColdFusion applications on that server. In Enterprise edition, multiple sandboxes, each with its own set of resource limitations may be configured. Applications may then be confined to these secure areas, thereby flexibly restricting the access that the application has to server resources.

Multiple Server Instances

All of the security techniques described thus far apply to each installation of a ColdFusion server. Each server has one ColdFusion Administrator, one RDS access configuration, one or more security sandboxes (depending on product edition), and so forth. In earlier versions, ColdFusion could only be installed once per server and all applications on that machine executed within a single server process. The J2EE configuration of Macromedia ColdFusion MX Enterprise 6.1, however, allows multiple ColdFusion server instances to run on a single physical server.

In this configuration, ColdFusion is installed on a J2EE server instance running within its own Java Virtual Machine (JVM). As its name implies, a virtual machine is a self-contained operating environment that behaves as if it were a separate computer while actually sharing resources with other virtual machines on the same server. As a result, a customer may install another J2EE server instance on another, completely separate JVM, and install ColdFusion there as well. These two ColdFusion servers then operate in complete isolation of each other—just as though they were on separate physical servers. This provides some powerful new security possibilities.

For those hosting applications for multiple departments or clients on the same server with previous versions of ColdFusion, granting developers access to the ColdFusion Administrator may have been a security risk if one developer inadvertently changed a setting that would impact someone else’s application. With ColdFusion MX Enterprise 6.1 deployed in multiple server instances, each site or application can have its own instance, and therefore its own ColdFusion Administrator. The owner of each ColdFusion instance can safely make changes within the Administrator that will not affect other instances. This allows resources such as custom tags, ColdFusion components, Java classes, and data sources to be safely isolated on a per-application basis.

Applications isolated within their own instance also guarantee the security of shared scopes such as server, application, and session scopes. Developers can use these shared scopes without concern that another application on the same server may overwrite an important value, or gain access to sensitive information stored in those scopes.

Finally, within each instance, ColdFusion applications can still take advantage of the various runtime security features discussed earlier. So even within the secure isolation of each server instance, particular sandbox security configurations, for example, allow further fine-grained control of application resources.

Conclusion

There are many valid security concerns that must be considered when building web applications. Solutions must be available to address these concerns in both application development and deployment phases. This brief demonstrates how ColdFusion MX 6.1 addresses the security concerns of snooping and eavesdropping, user impersonation, and unauthorized access with powerful, yet easy to implement data encryption, user authentication, and access control features. Resources where you can find additional information on best practices and instructions on implementing the features discussed here are listed in the Appendix.

Appendix

Disabling RDS access

To disable the RDSServlet mapping:

1. Back up the web.xml file.
This file is in the *cf_root*\wwwroot\WEB-INF directory on Windows and in the *cf_root*/wwwroot/WEB-INF directory on UNIX.
2. Open the original web.xml file for editing.
3. Comment out the RDSServlet mapping, as shown in the following example:

```
<!--  
  
<servlet-mapping>  
  
<servlet-name>RDSServlet</servlet-name>  
  
<url-pattern>/CFIDE/main/ide.cfm</url-pattern>  
  
</servlet-mapping>
```

-->

4. Save the file.
5. Restart ColdFusion MX.

Additional ColdFusion Security Resources

Macromedia maintains a site called the Security Zone which periodically publishes security bulletins and technical briefs that provide information to customers about security issues believed to be significant. The Security Zone may be accessed at http://www.macromedia.com/devnet/security/security_zone/.

Macromedia also provides a Security Notification Service, a free e-mail notification service that Macromedia uses to send information to customers about the security of Macromedia products. Bulletins are sent describing any known security issue, its impact, and how customers can protect themselves. The bulletins also detail any additional actions Macromedia plans to take and additional resources that may be available. Anyone may subscribe or unsubscribe [here](http://www.macromedia.com/devnet/security/security_zone/notification_service.html).

