



安全性是 Adobe、用户、系统管理员和应用程序开发人员关注的焦点。因此，Adobe® AIR™ 包含一组安全性规则和控制，以保护用户和应用程序开发人员的利益。本白皮书介绍了在使用和开发 Adobe AIR 的应用程序时的一些安全注意事项。

概述 1

安装和更新应用程序 2

代码签名 3

安全沙箱 4

访问文件系统 5

安全使用不受信任的内容 6

HTML 安全性 6

其它安全注意事项 7

概述

虽然 AIR 安全模型是由用于 Flash® Player 中运行的 SWF 内容和浏览器中运行的 HTML 内容的安全模型发展而来的，但此安全协议与适用于浏览器中内容的安全协议不同。此协议为开发人员提供了一种自由访问更广泛的功能来获得丰富体验的安全方式，这些功能不适用于基于浏览器的应用程序。

运行 AIR 应用程序需要的用户权限与运行本机应用程序需要的用户权限相同。通常，使用这些权限可以对操作系统功能进行广泛的访问，例如读取和写入文件、绘制到屏幕以及与网络进行通信。操作系统中适用于本机应用程序的限制（例如特定于用户的权限）同样适用于 AIR 应用程序。

AIR 应用程序是采用编译过的字节代码（SWF 内容）或解释过的脚本（JavaScript、HTML）编写的，以便运行时提供内存管理。这样可以最大程度地减少与内存管理（如缓冲区溢出和内存损坏）有关的漏洞对 AIR 应用程序产生影响的可能性。下面是一些影响用本机代码编写的桌面应用程序的最常见漏洞。

注意：本白皮书将讨论 AIR 运行时中与安全相关的问题。有关开发安全的 AIR 应用程序的技术细节和使用 AIR API 的注意事项，请参阅开发人员文档中的“[AIR 安全性](#)”一章：

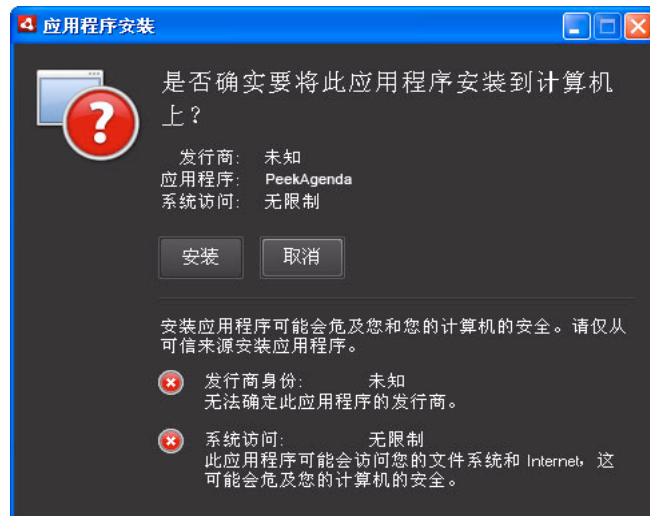
- 对于 Flex 开发人员 (http://www.adobe.com/go/learn_air_flex_cn)
- 对于 Flash 开发人员 (http://www.adobe.com/go/learn_air_flash_cn)
- 对于 Ajax 开发人员 (http://www.adobe.com/go/learn_air_html_cn)

安装和更新应用程序

AIR 应用程序是通过扩展名为 `air` 的 AIR 安装程序文件分发的。如果安装了 AIR 运行时且打开了 AIR 文件，运行时将管理应用程序的安装过程。

注意：开发人员可以指定版本、应用程序名称和发行商来源，但初始应用程序安装流程本身无法修改。此限制对用户非常有利，因为所有 AIR 应用程序共享由 AIR 运行时管理的安全、流畅和一致的安​​装过程。如果有必要对应用程序进行自定义，则可以在首次执行应用程序时进行自定义。

默认的应用程序安装程序为用户提供了与安全相关的信息。如果 AIR 应用程序已使用可信证书签名，或者所使用的签名证书链接到安装计算机上的可信证书，则 AIR 在安装过程中将显示发布者名称。否则，发布者名称将显示为“未知”。因此，用户可以对是否安装应用程序做出明智的决定：



AIR 应用程序首先要求用户的计算机上安装了运行时，就像 SWF 文件首先要求安装了 Flash Player 浏览器插件一样。

可以通过两种方式安装运行时：使用无缝安装功能或通过手动安装。

- 利用无缝安装功能，可以为没有安装 Adobe AIR 的开发人员提供流畅的安装体验。在无缝安装方法中，开发人员可以将 SWF 文件嵌入到网页中，该 SWF 文件会提供安装的 AIR 应用程序的名称。用户单击该 SWF 文件安装应用程序时，SWF 文件会检查是否存在 AIR 运行时。如果无法检测到运行时，运行时会自动安装并被开发人员的应用程序的安装过程立即激活。用户可以选择取消安装。
- 用户也可以在安装 AIR 文件之前手动下载并安装运行时。开发人员随后可以通过不同的方式（例如通过电子邮件或网站上的 HTML 链接）分发 AIR 文件。打开 AIR 文件后，运行时即被激活并开始处理应用程序的安装。

用户可以使用 AIR 安全模型决定是否安装 AIR 应用程序。AIR 安装程序对本机应用程序安装技术提供了若干改进，使得用户更方便地做出以下信任决策：

- 即使通过 Web 浏览器中的链接安装 AIR 应用程序，运行时也会在所有操作系统上提供一致的安裝体验。大多数本机应用程序安裝体验根据浏览器或其它应用程序提供安全信息（如果提供了安全信息）。
- AIR 应用程序安裝程序可以确定应用程序的源（即，如果无法验证源，安裝程序会对此加以澄清），并提供有关应用程序可用权限的信息（如果用户允许继续安裝）。
- 运行时會管理 AIR 应用程序的安裝过程。AIR 应用程序无法处理运行时使用的安裝过程。

通常，用户不应安裝来自其不信任源或无法验证源的任何桌面应用程序（包括 AIR 应用程序）。对本机应用程序执行的安全验证同样适用于 AIR 应用程序，因为安全验证适用于其它可安裝应用程序。

更新 AIR 应用程序

开发和部署软件更新是本机代码应用程序面临的最大的安全挑战之一。安裝的 AIR 应用程序可以检查更新 AIR 文件的远程位置。如果存在适当的更新，则会下载并安裝 AIR 文件，然后重新启动该应用程序。有关使用此方法提供新功能和响应潜在安全漏洞的详细信息，请参阅开发人员文档。

卸载 AIR 应用程序

用户可以卸载 AIR 应用程序：

- 在 Windows 中：使用“添加 / 删除程序”面板删除该应用程序。
- 在 Mac OS 中：从安裝位置删除 app 文件。

删除 AIR 应用程序将删除该应用程序目录中的所有文件。但不会删除应用程序可能已写入应用程序目录外部的文件。删除 AIR 应用程序不会撤消 AIR 应用程序对该应用程序目录外部的文件所做的更改。

代码签名

AIR 运行时要求所有 AIR 应用程序都需要进行数字签名。代码签名是一种数字签名代码过程，用于确保软件和发布者身份的完整性。开发人员可以通过证书颁发机构 (CA) 颁发的证书或构建自签名证书对 AIR 应用程序进行签名。

使用公认的证书颁发机构 (CA) 颁发的证书对 AIR 文件进行数字签名，可以向用户提供重要保证：他们正在安裝的应用程序没有被意外或恶意更改。使用公认的证书颁发机构 (CA) 颁发的证书对 AIR 文件进行数字签名还可以将开发人员标识为签名者（发布者）。AIR 可识别由 Verisign 和 Thawte 证书颁发机构颁发的代码签名证书。如果开发人员已使用 Verisign 或 Thawte 证书对 AIR 文件进行签名，则 AIR 应用程序安裝程序在安裝过程中将显示发布者名称。

如果 AIR 应用程序已使用可信证书签名，或者所使用的签名证书链接到安裝计算机上的可信证书，则 AIR 应用程序安裝程序在安裝过程中将显示发布者名称。证书颁发机构 (CA) 在颁发具有高可靠性的证书之前，将使用完善的验证过程对发布者或开发人员的身份进行验证。

开发人员还可以使用他们自己创建的自签名证书对 AIR 应用程序进行签名。但是，AIR 应用程序安装程序会将这些应用程序表示为来自未经认证的发布者。

AIR 文件签名后，安装文件中将包含数字签名。签名包括包的摘要（用于验证 AIR 文件自签名后未被更改）和有关签名证书的信息（用于验证发布者身份）。

AIR 使用的公钥基础结构 (PKI) 通过操作系统的证书存储支持。安装 AIR 应用程序的计算机必须直接信任用于对 AIR 应用程序签名的证书，或者该计算机必须信任将证书链接到可信证书颁发机构的证书链，以便验证发布者信息。

如果对 AIR 文件签名的证书并未链接到一个可信根证书（这通常包括所有自签名证书），则无法验证发布者信息。尽管 AIR 可以确定该 AIR 文件自签名后没有被更改，但无法验证实际创建和签署该文件的人员。

有关代码签名过程和认可的证书格式的详细信息，请参阅开发人员文档。

安全沙箱

AIR 提供了一个全面的安全体系结构，用于定义 AIR 应用程序中每个文件的权限。这包括随应用程序一起安装的那些文件和应用程序加载的其它文件。根据文件的来源授权其相应的权限，并将其分配到称为沙箱的逻辑安全组中。

随应用程序一起安装的文件在称为“应用程序目录”的目录中，因此，在默认情况下，这些文件将放在称为“应用程序沙箱”的安全沙箱中，该沙箱拥有访问所有 AIR API 的权限。如果其中某些 API 可供除应用程序资源目录之外的源中的内容（即那些没有随应用程序一起安装的文件）使用，则会带来巨大安全风险。

沙箱的 AIR 安全模型由 Flash Player 安全模型以及应用程序沙箱组成。不在应用程序沙箱中的文件具有的安全限制与 Flash Player 安全模型指定的安全限制类似。

运行时使用这些安全沙箱定义文件可以访问的数据范围以及可以执行的操作。若要维护本地安全，请将各个沙箱中的文件进行隔离。例如，从外部 Internet URL 加载到 AIR 应用程序的 SWF 文件放置在远程沙箱中，该文件默认情况下不具有通过脚本访问应用程序目录中分配给应用程序沙箱的文件的权限。

应用程序沙箱中的内容的权限

安装应用程序时，AIR 安装程序文件中包括的所有文件都会安装到用户计算机的应用程序目录中。在应用程序运行时，应用程序目录树中的所有文件都会分配到应用程序沙箱中。应用程序沙箱中的内容具有 AIR 应用程序的完全访问权限，包括与本地文件系统内容进行交互。

许多 AIR 应用程序只能使用这些本地安装的文件来运行应用程序。但是，不会限制 AIR 应用程序仅加载应用程序目录中的文件，它们可以加载任意源中任何类型的文件。这包括用户计算机上的文件以及外部源中的文件（例如本地网络或 Internet 上的文件）。文件类型不会对安全限制产生影响；加载的 HTML 文件与从相同源加载的 SWF 文件具有相同的安全权限。（但是，限制应用程序沙箱中的内容加载沙箱外的 JavaScript 文件。详细信息请参阅开发人员文档。）

应用程序安全沙箱中的内容可以访问 AIR API，而其它沙箱中的内容则无法访问。例如，只有应用程序安全沙箱中的内容才可以读取和写入本地文件系统。

有多种 JavaScript 技术可将字符串动态转换为可执行代码。从应用程序 URL 加载代码时，应用程序安全沙箱中的内容只能使用这些技术。在应用程序沙箱中使用这些技术会带来安全风险。例如，应用程序可能会在无意中执行从网络沙箱加载的字符串，而该字符串可能包含恶意代码，如删除或更改用户计算机上的文件或者将本地文件内容报告回不受信任的网络域的代码。详细信息请参阅开发人员文档。

非应用程序沙箱中的内容的权限

从网络或 Internet 位置加载的文件会分配到非应用程序沙箱。这些内容在行为方式上具有一组与 Web 浏览器中运行的 SWF 内容（在 Flash Player 中）或 Web 浏览器中运行的 HTML 内容相同的权限和限制。

与应用程序安全沙箱中的内容不同，非应用程序安全沙箱中的 HTML 代码在任何时候都可以使用 JavaScript 方法执行动态生成的代码。

非应用程序沙箱中的代码无权访问提供应用程序功能的受权限保护的 AIR API。

详细信息请参阅开发人员文档。

访问文件系统

在 Web 浏览器中运行的应用程序只能与用户的本地文件系统有限的交互。Web 浏览器会实施安全策略，用于确保用户的计算机不会由于加载 Web 内容而被破坏。例如，通过 Flash Player 在浏览器中运行的 SWF 文件无法直接与已经在用户计算机中的文件进行交互。可以将共享对象写入用户的计算机，以便维护用户首选参数和其它数据，但文件系统交互将受到此限制。由于 AIR 应用程序安装在本地，因此它们与最终用户之间有不同的安全协议。应用程序与最终用户之间的这一协议是在安装时建立的，就像本机应用程序一样，其中包括应用程序在本地文件系统中读取和写入的功能。

这一灵活性要求开发人员担负较高的责任。意外的应用程序安全漏洞不仅会危害应用程序的功能，而且会危害用户计算机的完整性。开发人员文档中的“[AIR 安全性](#)”一章介绍了有关的最佳做法。

除非用户计算机应用了管理员限制，否则 AIR 应用程序有权写入用户硬盘上的任意位置。但是，建议开发人员使用运行时为每个应用程序提供的特定于用户和特定于应用程序的应用程序存储目录。利用 AIR API，开发人员可以方便地读取应用程序存储目录中的数据和向其中写入数据。该运行时还为每个应用程序和用户提供唯一的加密本地数据存储区域。这允许应用程序以加密格式保存和检索存储在用户本地硬盘上的数据，其他应用程序或用户无法解密这些数据。独立的加密本地存储用于每个 AIR 应用程序，每个 AIR 应用程序对每个用户使用独立的加密本地存储。应用程序可以使用加密的本地存储来存储必须保护的信息，如用于 Web 服务的登录凭据。AIR 使用 DPAPI（在 Windows 上）和 KeyChain（在 Mac OS 上）将加密的本地存储与每个用户相关联。加密的本地存储使用 AES-CBC 128 位加密。

安全使用不受信任的内容

未分配给应用程序沙箱的内容可以为 AIR 应用程序提供其它脚本功能，但前提是满足运行时的安全条件。本节介绍 AIR 安全协议以及非应用程序内容。

AIR 应用程序限制脚本访问非应用程序内容比 Flash Player 浏览器插件限制脚本访问不受信任内容要严格的多。例如，在浏览器的 Flash Player 中，SWF 文件可以调用 `System.allowDomain()` 方法，对从指定域加载的任何 SWF 文件授予脚本访问权限。AIR 应用程序安全沙箱中的内容不允许调用此方法，以防止向非应用程序文件授予对用户文件系统的不合理访问权限。

通过脚本访问应用程序和非应用程序内容的 AIR 应用程序具有更复杂的安全安排。只允许应用程序沙箱外的文件使用沙箱桥来访问应用程序沙箱中的文件的属性和方法。沙箱桥充当应用程序内容与非应用程序内容之间的通道，在两个文件之间提供显式交互。如果使用正确，沙箱桥会提供额外的安全层，从而限制非应用程序内容访问属于应用程序内容的对象引用。

通过示例可以更好地说明沙箱桥的优点。假设 AIR 音乐商店应用程序需要为希望创建自己的 SWF 文件的广告商提供 API，商店应用程序可以使用这些文件进行通信。该商店需要为广告商提供在商店中查找艺术家和光盘的方法，另外出于安全原因，还需要将某些方法和属性与第三方 SWF 文件进行隔离。

沙箱桥可以提供此功能。默认情况下，在运行时从外部加载到 AIR 应用程序的内容无权访问主应用程序中的任何方法或属性。通过自定义沙箱桥，开发人员可以在不公开这些方法或属性的情况下为远程内容提供服务。沙箱桥在受信任和不受信任内容之间架起了一个有限通道。

有关使用沙箱桥的完整信息，请参阅开发人员文档中的“[AIR 安全性](#)”一章。

HTML 安全性

HTML 内容的安全注意事项不同于基于 SWF 的内容，主要是因为 JavaScript 能够创建动态生成的代码。如果在应用程序沙箱内允许使用动态生成的代码（如调用 `eval()` 函数时生成的代码），则会带来安全风险。例如，应用程序可能会在无意中执行从网络沙箱加载的字符串，而该字符串可能包含恶意代码，如删除或更改用户计算机上的文件或者将本地文件内容报告回不受信任的网络域的代码。

生成动态代码的方式如下所示：

- 调用 `eval()` 函数。
- 设置 `innerHTML` 属性或调用 DOM 函数以插入脚本标签，从而直接加载资源外部的脚本。
- 设置 `innerHTML` 属性或调用 DOM 函数以插入具有内联代码的脚本标签（而非通过 `src` 加载脚本）。
- 将应用程序沙箱中内容的 `script` 标签的 `src` 设置为一个不在应用程序资源目录中的文件。
- 使用 javascript URL 架构（如 `href="javascript:alert('Test')"` 所示）。

从应用程序目录加载内容时，应用程序安全沙箱中的代码只能使用这些方法。这可以防止应用程序沙箱（具有对所有 AIR API 的访问权限）中的代码执行可能来自不受信任源中的脚本。

非应用程序安全沙箱中的内容可以使用这些方法生成动态代码。但是，它们不能直接访问 AIR API。AIR 沙箱桥功能为非应用程序安全沙箱中的代码提供了一种与应用程序沙箱中的代码进行交互的方式（交互方法由应用程序代码限制和决定）。

- 对于 Flex 开发人员 (http://www.adobe.com/go/learn_air_flex_cn)
- 对于 Flash 开发人员 (http://www.adobe.com/go/learn_air_flash_cn)
- 对于 Ajax 开发人员 (http://www.adobe.com/go/learn_air_html_cn)

有关 HTML 安全性的详细信息，请参阅开发人员文档中“**AIR 安全性**”一章。

另外，请参阅《Adobe AIR HTML 安全性》白皮书（发布于 http://www.adobe.com/go/learn_air_htmlsecurity_wp_cn）。

其它安全注意事项

虽然 AIR 应用程序是使用 Web 技术构建的，但开发人员应知道这些应用程序并不在浏览器安全模型中运行，这一点很重要。这意味着，有可能构建会对本地系统有意或无意产生损害的 AIR 应用程序。AIR 会尝试最大程度降低此风险，但仍存在一些可能引入漏洞的方式。本节介绍了重要的潜在不安全因素。有关构建避免这些风险的应用程序的最佳做法，请参阅开发人员文档。

将文件导入应用程序安全沙箱的风险

应用程序沙箱中的内容具有运行时的完全权限。建议开发人员考虑使用以下方法：

- 仅在必要时才在 AIR 文件（位于安装的应用程序中）中包含文件。
- 仅在完全理解和信任脚本文件的行为时才在 AIR 文件（位于安装的应用程序中）中包含该脚本文件。
- 不要将来自网络的数据用作可能导致代码执行的 AIR API 的方法的参数。

Adobe AIR 可防止应用程序沙箱中的内容将来自网络的数据作为代码使用，以免无意中执行恶意代码。这包括使用 ActionScript `Loader.loadBytes()` 方法和 JavaScript `eval()` 函数。

使用外部源确定路径的风险

使用外部数据或内容时可能会破坏 AIR 应用程序。因此，应用程序在使用网络或文件系统中的数据时应特别小心。信任责任最终取决于开发人员以及他们建立的网络连接，但加载外来数据本身具有风险，不应将其用于输入敏感操作。建议开发人员不要使用以下方法：

- 使用来自网络的数据确定文件名
- 使用来自网络的数据构建应用程序用来发送私人信息的 URL

使用、存储或传输不安全凭据的风险

将用户凭据存储在用户的本地文件系统中本身会带来可能破坏这些凭据的风险。建议开发人员考虑使用以下方法：

- 如果凭据必须存储在本地，请在写入本地文件系统时加密凭据。AIR 运行时提供了对每个安装的应用程序都具有唯一性的加密存储，开发人员文档中对此进行了详细介绍。
- 不要将未加密的用户凭据传输到网络源中，除非该源受信任。
- 决不要在创建凭据时指定默认密码，应让用户自己创建密码。保留默认值的用户会将其凭据公开给已经知道默认密码的攻击者。

降级攻击的风险

在安装应用程序过程中，运行时会进行检查以确保应用程序的版本不是当前安装的版本。如果应用程序已经安装，则运行时会将现有应用程序的版本字符串与正在安装的版本进行比较。如果此字符串不同，则用户可以选择升级安装。运行时无法保证新安装的版本比旧版本新，仅保证版本不同。攻击者可能会向用户分发旧版本以避免安全漏洞。采取相应措施可以降低这种风险，开发人员文档提供了实现规避风险的版本架构和更新检查的最佳做法。