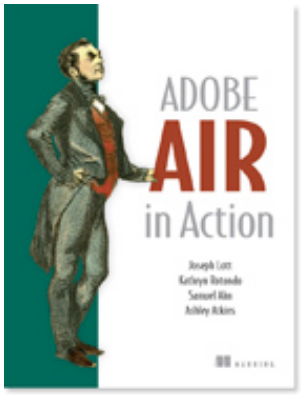
MANNING PUBLICATIONS

AIR application security and authenticity

Based on



Adobe Air in Action

EARLY ACCESS EDITION

Joey Lott, Kathryn Rotondo, Sam Ahn and Ashley Atkins
Print book: July 2008 | 344 pages
ISBN: 1933988487

This article is based on chapter 1 from Adobe Air in Action *by Joey Lott, Kathryn Rotondo, Sam Ahn and Ashley Atkins. For more information, please visit www.manning.com/lott.*

Any discussion of Adobe AIR would be remiss without a discussion of two related issues: security and authenticity. These are two issues that are important for you to consider as an application developer because any breaches or violations would reflect poorly on you. Therefore, it's important that you have a good understanding of what AIR does and does not enforce in the way of application security and authenticity and what steps you need to take to protect users of your applications.

1. Understanding AIR Application Security

One of Adobe's flagship products is Flash Player, a product that has been so successful, in part, because of the great lengths Adobe (and previously Macromedia) has gone to in order to ensure that Flash developers cannot intentionally or unintentionally cause harm to a user's computer system. Flash Player has a lot of security features to protect users. This gives users peace of mind when viewing Flash content on the Web. Users know that the Flash content will not cause problems for their computer system.

AIR applications are desktop applications, and as such it is essential that they have greater access to the user's computer system than Web-based Flash applications. Even though AIR applications can run Flash content, that Flash content has more opportunity to harm the user's system than Web-based Flash content. It's a trade-off: a vastly greater feature set, but increased risk as well.

AIR applications still run through a mediator – the runtime environment itself. Therefore, Adobe still has a great deal of control over just what an AIR application can and cannot do.

However, while many risks are mitigated by the runtime environment, AIR still allows applications many more privileges than their Web counterparts might have.

The first thing that you as an AIR developer must be aware of is that it is incumbent on you to treat the users of you application with great respect by taking security matters seriously. For example, it is important that you closely manage all parameters to code that might run in your application. Do not allow the user to arbitrarily enter values and do not use dynamic, network-originating values as parameters for code that can do things such as access the file system. You can read a more detailed security whitepaper from Adobe at download.macromedia.com/pub/labs/air/air_security.pdf.

2. Ensuring application authenticity

In order to help users of your application have peace of mind Adobe requires that all AIR applications are digitally signed. (Note that signing is only necessary to build the installer, and you can still build and test your AIR applications without a signature of any sort.) A digital signature helps to potentially verify two things to the user: authenticity and integrity. A digital signature is meant to mimic a traditional hand-written signature of ink on paper in that it verifies the publisher of the application (authenticity) and that it has not been altered since it was published (integrity).

You can prove that AIR enforces integrity if you'd like with a simple test. What you can do in this experiment is verify that by modifying the .air file the AIR runtime will refuse to run the installation. All that you need is an .air file and zip utility. The .air format is an archive format that any zip utility can read. Do the following.

1. Run the .air file to verify that the AIR runtime will initially prompt you to run the installation. You don't need to actually click the install button from the wizard once it appears. All you need to verify is that the AIR runtime will give you the option to install.
2. Click the cancel button to exit the install wizard.
3. Use a zip utility to add a file to the archive. Any file will work. For the purposes of this exercise you can create a new blank text file and add it to the archive. If you are on a Windows computer the simplest way to achieve this is to change the .air file extension to .zip, drag the text file into the .zip archive, and then change the file extension back to .air.
4. Run the .air file. This time you'll receive an error message saying that the .air file is damaged and cannot be installed.

For AIR applications digital signatures appear together with digital certificates. There are two basic types of certificates: self-signed certificates and certificates issues by certification authorities. There are advantages and disadvantages to each.

Self-signed certificates are advantageous in that they are the easiest to procure. The Flash CS3 AIR update and the Flex 3 SDK (and subsequently Flex Builder 3) provide mechanisms for creating self-signed certificates for your AIR applications. You can read about the details of how to create these types of certificates later in this chapter. Self-signed certificates do provide a level of security to users in that they verify the integrity of the application. However, they do little to nothing to assure users about the authenticity of the publisher. It's a bit like you acting as a notary for your own documents. As a result, Adobe displays the publisher identity as unknown in the installation wizard for self-signed certificates. This is clearly disadvantageous because it doesn't create a feeling of security for users, and they are less likely to opt to install an application from an unknown publisher than they would be if the identity of the publisher could be verified.

Certification authorities are organizations that issues certificates and act as a third party to verify your identity. Certification authorities issue certificates only once they have verified your identity, usually by requesting documents such as government-issued IDs. The advantage of certificate issued by a certification authority is that gives more assurance of your actual identity than a self-signed certificate. As a result Adobe displays the identity listed in the certificate as the publisher identity in the installation wizard for certificates issued by a certification authority. On the other hand, it might be obvious what some of the disadvantages are: obtaining a certificate from a certification authority is more difficult and requires more time than a self-signed certificate. Also, be aware that most certification authorities charge a fee for certificates (at the time of this writing the largest issue charges $299 USD for a code signing certificate for an AIR application.)

Two of the most well-known certificate issuers are VeriSign (www.verisign.com) and Thawte (www.thawte.com) (technically Thawte is now owned by VeriSign.) If you want to provide the highest level of certification for your AIR application you'll need to purchase a certificate from one of these issuers. You'll need what is called a code signing certificate. You can find more information about purchasing a certificate from the Web sites of the issuers.

NOTE

There are certification authorities other than Verisign and Thawte, and there are even non-commercial certification authorities such as cacert.org that grant code signing certificates. However, do your research before purchasing or otherwise acquiring a certificate (cacert.org still requires that you do a fair amount of leg work to obtain a code signing certificate) in order to make sure that the certificate will be trusted on the majority of computers. If the certificate is not trusted then the publisher of the AIR application will still show up as unknown. Speak to someone at the organization that grants the certificates and ask questions if you are in doubt.

When getting started building AIR applications you'll probably be hesitant to invest in purchasing a certificate just to put together a few examples and send the installers to your friends. Again, remember that the certificate is only necessary when you want to create the installer. You can always test AIR applications without a certificate.

However, when you're ready to create an .air file for your application you'll need to give careful consideration to how you want to digitally sign the application. You can only associate a certificate with an application once. That means you cannot use a self-signed certificate initially and then change to a certificate from a certification authority later on. If you resign with a different certificate then users of earlier versions of the application will not be able to upgrade.

3. Conclusion

When you create AIR applications you are creating a new type of application that is neither entirely an Internet application nor is it entirely a desktop application. Instead, AIR applications blend the best of both worlds. However, along with the power of building AIR applications you also must take on added responsibility for providing application security and guaranteeing application authenticity. To verify application authenticity all AIR applications must be digitally signed with a certificate as you learned in this article. To learn more about Adobe AIR you can read AIR in Action from Manning Publications.