Adobe Cross Domain Policy File Specification

Version 2.0

Adobe Systems, Inc.
Modification date: 8/2/10

Adobe® Cross Domain Policy File Specification V. 2.0

Contents

Cross-domain Policy File Specification

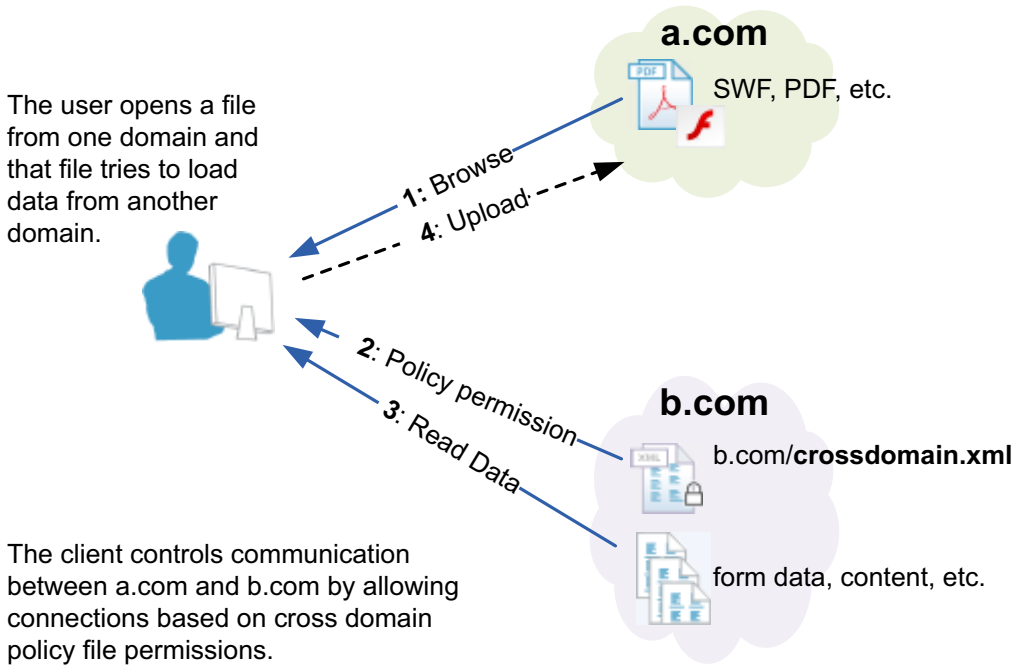
1.1 Introduction

A *cross-domain policy file* is an XML document that grants a web client, such as Adobe Flash Player or Adobe Acrobat (though not necessarily limited to these), permission to handle data across domains. When a client hosts content from a particular source domain and that content makes requests directed towards a domain other than its own, the remote domain needs to host a cross-domain policy file that grants access to the source domain, allowing the client to continue the transaction.

1.1.1 Typical workflow

In Figure 1, b.com hosts a policy file that grants permission for a file on a.com to load data from some or all of the b.com site depending on the policy file's location and configuration.

Figure 1 Cross domain workflow



The b.com administrator should exercise care because policy files give permission for files from other domains to operate with users' credentials when accessing b.com. In the diagram above, the user browses to a file on a.com (**1**). The file from a.com obtains permission from b.com's policy file (**2**) to load data from b.com (**3**). If b.com hosts a policy and requires credentials for access then any documents served from the domains listed in b.com's policy file gain the right to use those credentials on the user's behalf. Now that the a.com file has access to data from b.com, the document can upload the b.com data back to its own server at a.com (**4**)

It is important here to compare a user's and a.com's server's access to b.com. If the user is able to access b.com because both are inside a network firewall or because a required login returned an HTTP cookie, then the user has greater privileges than the a.com server. In this situation, it may be risky to create a policy file on b.com because files from a.com must then be trusted not to disclose data that is private to b.com, the user, or both.

Given the potential of policy files to create this kind of vulnerability, an administrator should limit the number and location of policy files. It should always be possible to easily audit the current set of permissions and control who can add and modify policy files.

1.1.2 Terminology

This document uses the following terminology:

- **Requesting domain**: The origin of the server requesting document.

  In Figure 1, this would be a.com. It is this first domain origin in a cross domain workflow that is compared against subsequent participating domains (for target content and target domains) to see if they conform to a same origin policy. If the origins differ, a cross domain policy is required.

- **Requesting document**: The document served by the requesting domain.

  In Figure 1, the user browses to the requesting document (**1**). The document requests target content from the target domain.

- **Target content**: The content requested by the requesting document that resides on a different domain.

  In Figure 1, the content requested from b.com by the a.com document that is only read if an appropriate policy file is present (**3**).

- **Target domain**: The target content's host.

  In Figure 1, the client checks the target domain for a policy file. If the requisite permissions exist, the requesting document access the target content.

1.2 Policy file definitions

The cross-domain policy file schema is available as a DTD (Document Type Definition) or XSD (XML Schema Definition).

1.2.1 Document type definition (DTD)

The DTD is as follows:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
   <!-- Adobe DTD for cross-domain policy files -->
   <!-- Copyright (c) 2008, Adobe Systems Inc. -->

<!ELEMENT cross-domain-policy

(site-control?,allow-access-from*,allow-http-request-headers-from*,allow-access-
from-identity*)>
```

```
<!ELEMENT site-control EMPTY>
<!ATTLIST site-control permitted-cross-domain-policies
   (none|master-only|by-content-type|by-ftp-filename|all) "none" #REQUIRED>

<!ELEMENT allow-access-from EMPTY>
<!ATTLIST allow-access-from domain CDATA #REQUIRED>
<!ATTLIST allow-access-from to-ports CDATA #IMPLIED>
<!ATTLIST allow-access-from secure (true|false) "true">

<!ELEMENT allow-http-request-headers-from EMPTY>
<!ATTLIST allow-http-request-headers-from domain CDATA #REQUIRED>
<!ATTLIST allow-http-request-headers-from headers CDATA #REQUIRED>
<!ATTLIST allow-http-request-headers-from secure (true|false) "true">

<!ELEMENT allow-access-from-identity (signatory)>

<!ELEMENT signatory (certificate)>

<!ELEMENT certificate EMPTY>
<!ATTLIST certificate fingerprint CDATA #REQUIRED>
<!ATTLIST certificate fingerprint-algorithm CDATA #REQUIRED>

<!-- End of file. -->
```

1.2.2 Schemas (XSDs)

Schemas describe the structure of a document. XSDs are available to define the generic policy file schema, as well as each different type of policy file (either HTTP, HTTPS, FTP, or Socket) since policy files hosted in each of those contexts are slightly different. Table 1lists all available schemas for policy files.

Table 1 Policy file schemas

Schema	URL
Generic DTD	http://www.adobe.com/xml/dtds/cross-domain-policy.dtd
Generic XSD	http://www.adobe.com/xml/schemas/PolicyFile.xsd
HTTP XSD	http://www.adobe.com/xml/schemas/PolicyFileHttp.xsd
HTTPS XSD	http://www.adobe.com/xml/schemas/PolicyFileHttps.xsd
FTP XSD	http://www.adobe.com/xml/schemas/PolicyFileFtp.xsd
Socket XSD	http://www.adobe.com/xml/schemas/PolicyFileSocket.xsd

1.3 Element specification

1.3.1 cross-domain-policy

The `cross-domain-policy` element is the root element for cross-domain policy files. It is a container for policy file definitions and has no attributes of its own.

Table 2 Tree location: cross-domain-policy

Feature	Details
Child of	None; cross-domain-policy is the root node
Child elements	• site-control • allow-access-from • allow-access-from-identity • allow-http-request-headers-from

1.3.1.1 Attributes (none)

1.3.2 site-control

`site-control` defines the meta-policy for the current domain. A meta-policy specifies acceptable domain policy files other than the master policy file located in the target domain's root and named crossdomain.xml.

> **Tip:** The `site-control` element is specific to master policy files; if used within a policy file that is not the master policy file, the `site-control` element will be ignored.

If a client is instructed to use a policy file other than that of the master policy file, the client must first check the master policy's meta-policy to determine if the requested policy file is allowed.

Table 3 Element details: site-control

Feature	Details
Child of	cross-domain-policy
Child elements	None.

1.3.2.1 Attributes

permitted-cross-domain-policies

Specifies the meta-policy. The default value is `master-only` for all policy files except socket policy files, where the default is `all`. Allowable values include:

- **none**: No policy files are allowed anywhere on the target server, including this master policy file.

  > **Tip:** In other words, the root cross-domain policy does not contain `allow-access-from` directives or the HTTP headers. A meta-policy of "none" prevents the use of any other policies that may be present even if the developer included them. It is invalid to have `allow-access-from` or a header policy within a root cross-domain policy file with a meta-policy of "none". In cases where an invalid policy has both a "none" setting and other directives, "none" takes precedence and no permissions are allowed on the site.

- **master-only**: Only this master policy file is allowed.
- **by-content-type**: [HTTP/HTTPS only] Only policy files served with Content-Type: text/x-cross-domain-policy are allowed.

- **by-ftp-filename**: [FTP only] Only policy files whose file names are crossdomain.xml (i.e. URLs ending in /crossdomain.xml) are allowed.
- **all**: All policy files on this target domain are allowed.

1.3.3 allow-access-from

`allow-access-from` grants a requesting domain access to read data from the target domain. For each requesting that is given permission, a new `allow-access-from` element is required, although multiple domains can be given access with one `allow-access-from` element by using a wildcard (*).

Note: An `allow-access-from` element in a non-master policy file can only grant a requesting domain access to data within its own directory and subdirectories, not a directory in a higher level of that domain.

Table 4 Element details: site-control

Feature	Details
Child of	cross-domain-policy
Child elements	None.

Clients may choose to handle cross domain data differently depending on the format of that data. For text, or other nonvisual data, a policy file may be used to determine if the client is allowed to load the data from the target domain.

Alternatively, some data, such as images or other visual data, may be loaded from a target domain without permission from a policy file. Just because images from the target domain can be displayed does not mean that other data from that domain can be accessed. Should that content request information about the remote data, the policy file would then be checked to allow or disallow that action.

Tip: For example, a web page can contains an iframe pointing to another site. That site's images will appear in the iframe. However, while the requesting site can choose to display content from the target, it cannot access any data without first checking the cross domain policy.

1.3.3.1 Attributes

`allow-access-from` may have the following attributes:

- **domain**: Specifies a requesting domain to be granted access. Both named domains and IP addresses are acceptable values. Subdomains are considered different domains. A wildcard (*) can be used to match all domains when used alone, or multiple domains (subdomains) when used as a prefix for an explicit, second-level domain name separated with a dot (.). Specific, individual domains require separate `allow-access-from` elements. For more information, refer to "Matching rules" on page 12.

Note: To grant access to requesting documents that have no associated domain, such as local files, a "*" wildcard must be specified for the `domain` attribute. This has the effect of allowing access by `any` requesting document from `any` origin. However, granting

access to domainless requesting documents is a dangerous practice, and you should only grant access to "*" if you are certain that the scope of the policy file does not host any access-controlled, personalized, or private data.

- **to-ports**: [Sockets only] A comma-separated list of ports or range of ports that a socket connection is allowed to connect to. A range of ports is specified through a dash (-) between two port numbers. Ranges can be used with individual ports when separated with a comma. A single wildcard (*) can be used to allow all ports.
- **secure**: [HTTPS and Sockets only, optional] Specifies whether access is granted only to HTTPS documents from the specified origin (`true`) or to all documents from the specified origin (`false`). If `secure` is not specified in an HTTPS policy file, it defaults to `true`. Using `false` in an HTTPS policy file is not recommended because this compromises the security offered by HTTPS; for example, allowing man-in-the-middle attacks to gain access to the HTTPS data protected by the policy file.

  In socket policy files, the default is `false`. It is only useful to specify `secure="true"` when the socket server is accepting connections from the local host since local socket connections are generally not at risk of man-in-the-middle attacks that could alter the `secure="true"` declaration.

1.3.4 allow-access-from-identity

`allow-access-from-identity` grants permissions based on cryptographic credentials rather than origin as does `allow-access-from`. The only mechanism defined thus far is to allow access by any document that has been digitally signed by a given party, where the party is identified by certificate.

Table 5 Element details: allow-access-from-identify

Feature	Details
Child of	cross-domain-policy
Child elements	Signatory: Identifies the signing party. Only one such element is allowed per `allow-access-from-identity` element.

1.3.4.1 signatory

One signatory element is used to contain the certificate element.

Table 6 Element details: Signatory

Feature	Details
Child of	allow-access-from-identity
Child elements	certificate: Identifies the signing party's certificate. Only one such element is allowed per `signatory` element.

1.3.4.2 certificate

Contains the certificate's identifying attributes.

Table 7 Element details: Certificate

Feature	Details
Child of	Signatory
Child elements	None.

1.3.4.3 Attributes

A `certificate` element must contain two attributes:

- **fingerprint-algorithm**: The hash algorithm used to compute the certificate fingerprint. The attribute value is case-insensitive. The only hash algorithm name defined at present is sha-1, but others are expected to be defined as implementations require.
- **fingerprint**: The fingerprint of the signing certificate, expressed as a string of hexadecimal digits. The attribute value is case-insensitive, and may contain spaces or colons, which are ignored. In the case of a sha-1 hash-algorithm, `fingerprint` must contain 40 hexadecimal digits.

1.3.5 allow-http-request-headers-from

`allow-http-request-headers-from` grants a requesting document from a requesting domain to send user-defined headers to the target domain. Whereas the `allow-access-from` element grants permission to pull data from the target domain, this tag grants permission to push data in the form of headers.

Note: An `allow-http-request-headers-from` element in a non-master policy file can only allow headers to be sent to URLs within the directory in which the policy file is defined and that directory's subdirectories.

Table 8 Element details: allow-http-request-headers-from

Feature	Details
Child of	cross-domain-policy
Child elements	None.

1.3.5.1 Attributes

`allow-http-request-headers-from` may contain the following attributes:

domain: Specifies requesting domain to be granted access. Both named domains and IP addresses are acceptable values. Subdomains are considered different domains. A wildcard (*) can be used to match all domains when used alone, or multiple domains (subdomains) when used as a prefix for an explicit, second-level domain name separated with a dot (.). Individual domains require separate `allow-access-from` elements.

headers: A comma-separated list of headers that the allowed requesting domain is permitted to send. A wildcard character (*) can be used to allow all headers or for header suffixes, allowing for headers that start with the same characters but end with different characters.

secure: [HTTPS only, optional] When false, allows an HTTPS policy file to grant access to a request coming from an HTTP source. The default is true, providing only HTTPS sources permission. Using false is not recommended.

1.4 Deploying policies

1.4.1 Policy file content (mime) types

Ideally, all URL policy files should be served with Content-Type: text/x-cross-domain-policy. This is the same content type that the by-content-type value of the permitted-cross-domain-policies attribute uses to determine validity of a policy file. Other content (or MIME) types include:

- text/*
- application/xml
- application/xhtml+xml

1.4.2 Policy file locations

Master policy files are located at a domain's root with the filename of crossdomain.xml (e.g. http://example.com/crossdomain.xml). When clients require a policy file, they look at the root by default. A domain should always host a master policy file to enforce its intended meta-policy.

If a client is instructed to load a policy file other than the master policy file, the client must still check the master policy file to ensure that the meta-policy defined by the master policy file (via the `site-control` element) permits the use of the originally requested policy file. Without a master policy file, it is left to the client to enforce the default behavior.

Instead of relying entirely on master policy files for meta-policies, clients may also decide to check for a X-Permitted-Cross-Domain-Policies header in documents to specify a meta-policy. In addition to the values acceptable in permitted-cross-domain-policies, this header may also use a value of none-this-response to indicate that the current document should not be used as a policy file despite other headers or its content.

Non-master policy files can only grant access to data within their own directory or subdirectories.

1.4.3 Requesting a policy from a socket server

When a socket server hosts a policy file, the client requests that file by sending a single-node XML document containing `<policy-file-request/>` to the server. The socket server responds with the necessary policy file so that the client can continue with the connection or close it if the policy file does not permit it.

Note: Policy files for sockets should always be socket-based. URL policy files from an HTTP source should not be used to provide permissions to socket connections.

1.5 Examples

1.5.1 Matching rules

The following rules are used in determining if a value in the domain attribute of the `allow-access-from` or `allow-http-request-headers-from` elements matches an actual domain name:

- Individual named domains or subdomains must match exactly.
- Explicit IP addresses do not match named domains, even if they refer to the same host.
- Domain wildcards, such as *.example.com, match both the domain alone and any subdomains.
- An overall wildcard, (*) allows access by all requestors and is not recommended. allow-all permissions should only be used when all content in the policy file's scope is completely public.

Note: Cross-domain redirects are not permitted for policy files. If a client encounters a cross-domain redirect while retrieving a policy file, it will ignore the policy file.

Table 9 Domain matching examples

Domain value	will match...	will not match...
www.example.com	http://www.example.com	http://example.com http://www.example.net http://www.adobe.com
*.example.com	http://example.com http://www.example.com http://deep.subdomain.example.com	http://www.example.net http://www.adobe.com
http://*.example.com	http://example.com http://www.example.com http://deep.subdomain.example.com	Any https domains
127.0.0.1	http://127.0.0.1	http://localhost http://127.0.0 http://127.0.0.2
www.example.*	No matches, invalid domain	None.

1.5.2 Typical policy

Policy files grant read access to data, permit a client to include custom headers in cross-domain requests, and grant permissions for socket-based connections. The most common location for a policy file on a server is in the root directory of a target domain with the filename crossdomain.xml (e.g. http://example.com/crossdomain.xml)—the default location that clients check when a policy file is required. Policy files hosted this way are known as *master policy files*. Example 1.1 shows a typical URL (i.e. non-socket) master policy file which allows access to the example.com root domain with and without the www subdomain as well as any subdomains.

Example 1.1: allow-access-from: Allowing access to root domains

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <site-control permitted-cross-domain-policies="master-only"/>
   <allow-access-from domain="*.example.com"/>
   <allow-access-from domain="www.example.com"/>
   <allow-http-request-headers-from domain="*.adobe.com" headers="SOAPAction"/>
</cross-domain-policy>
```

The `site-control` element here specifies that only this master policy file should be considered valid on this domain. Below that, the `allow-access-from element` specifies that content from the example.com requesting domain can access any data within the target domain (the domain in which this policy file has been saved). Finally, the `allow-http-request-headers-from` element indicates that a `SOAPAction` header is also allowed to be sent with requests made to the target domain from adobe.com.

1.5.3 Permissive vs. restrictive policies

The following is the least permissive master policy file definition. It enforces a meta-policy that restricts any policy file, including this one, from granting permissions of any type to any requesting domain:

Example 1.2: cross-domain-policy: Least permissive policy

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <site-control permitted-cross-domain-policies="none"/>
</cross-domain-policy>
```

The following is the most permissive master policy file definition (*strongly not recommended*). It allows any policy file on the target domain to grant permissions, allows access to any of its file, and permits any header to be sent to the server, all of this possible even through HTTPS despite the source being HTTP:

Example 1.3: cross-domain-policy: Least restrictive policy

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <site-control permitted-cross-domain-policies="all"/>
   <allow-access-from domain="*" secure="false"/>
   <allow-http-request-headers-from domain="*" headers="*" secure="false"/>
</cross-domain-policy>
```

1.5.4 Meta vs. master policies

While Example 1.4 does not allow data access to this target domain, it does define a meta-policy that allows other policy files within this domain to determine how access is handled. In this case, the client is instructed to look for a policy file other than the master for permissions related to this domain.

Example 1.4: Meta policy: Allowing non-master policy files

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <site-control permitted-cross-domain-policies="by-content-type"/>
</cross-domain-policy>
```

Example 1.5 defines a meta-policy that allows only this master policy file to function for this target domain. It allows access to data on example.com and all of its subdomains:

Example 1.5: Meta policy: allowing only a master policy

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <site-control permitted-cross-domain-policies="master-only"/>
   <allow-access-from domain="*.example.com"/>
</cross-domain-policy>
```

1.5.5 HTTP-HTTPS communications

Example 1.6 demonstrates the most permissive use of `allow-access-from` granting any other domain access to the files on this target domain, even if an HTTP source is accessing data on this domain through HTTPS. It should be obvious that this practice is not recommended.

Example 1.6: allow-access-from: Most permissive access

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <allow-access-from domain="*" secure="false"/>
</cross-domain-policy>
```

1.5.6 Socket permissions

Example 1.7 should be served to a client through a socket connection when requested with policy-file-request. It permits access to content from example.com or any of its subdomains through ports 507 and any port between 516 and 523 (including 516 and 523).

Example 1.7: allow-access-from: Socket policy

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <allow-access-from domain="*.example.com" to-ports="507,516-523"/>
</cross-domain-policy>
```

1.5.7 Credential-based permissions

Example 1.8: allow-access-from-identity: Required configuration

```
<allow-access-from-identity>
  <signatory>
    <certificate
      fingerprint="01:23:45:67:89:ab:cd:ef:01:23:45:67:89:ab:cd:ef:01:23:45:67"
      fingerprint-algorithm="sha-1"/>
  </signatory>
</allow-access-from-identity>
```

1.5.8 Header-based permissions

Example 1.9 demonstrates how to allow any requesting domain to send the `SOAPAction` header to this target domain.

Example 1.9: allow-http-request-headers-from: Header usage with SOAPAction

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <allow-http-request-headers-from domain="*" headers="SOAPAction"/>
</cross-domain-policy>
```

Example 1.10 allows the Authorization header and any header beginning with the characters X-Foo from www.example.com to be sent to this target domain. If a request is coming from foo.example.com, only headers beginning with the characters X-Foo are allowed, not Authorization:

Example 1.10: allow-http-request-headers-from: Header usage with wildcard

```
<?xml version="1.0"?>
<!DOCTYPE cross-domain-policy SYSTEM
"http://www.adobe.com/xml/dtds/cross-domain-policy.dtd">

<cross-domain-policy>
   <allow-http-request-headers-from domain="www.example.com"
headers="Authorization,X-Foo*"/>
   <allow-http-request-headers-from domain="foo.example.com" headers="X-Foo*"/>
</cross-domain-policy>
```

1.5.9 Friendly names and alias'

Any alias or friendly name can be used as long as your DNS server can resolve these names:

```
<allow-access-from domain="myalias.com" />
<allow-access-from domain="myfriendlyname" />
```

1.5.10 IP address

If you specify an IP address, access is granted only to files loaded from that IP address using IP syntax (for example, http://65.57.83.12/document.pdf), not those loaded using domain-name syntax. Acrobat does not perform reverse DNS lookup. The following lines enable access only to `105.216.0.40`.

```
<allow-access-from domain="105.216.0.40" />
```

1.6 Product Implementations

1.6.1 Flash

Adobe Flash Player has supported the use of cross-domain policy files since Flash Player 6.

Table 10 Policy file support by version: Flash

Version	Feature
10,0,12,0	site-control's permitted-cross-domain-policies default for non-socket policy files is "master-only"
9,0,124,0 8,0,42,0	allow-http-request-headers-from element
9,0,115,0	site-control element (permitted-cross-domain-policies default is "all")
7,0,19,0	• Socket-based policy files • Use of policy-file-request
7,0,14,0	• Exact domain matching where subdomains are considered separate domains • allow-access-from's secure attribute and a separation of HTTPS from HTTP
6,0,21,0	• Basic cross-domain policy file support with allow-access-from element (domain and to-ports attributes) • Policy files for sockets were hosted through HTTP only (URL policy files)

1.6.1.1 Unsupported features

The following is not supported:

- `allow-access-from-identity` elements.

1.6.2 Acrobat

The Acrobat family of products has supported the use of cross-domain policy files since version 9.0.

Table 11 Policy file support by version: Acrobat and Adobe Reader

Version	Change
9.1	Support for allowing cross domain access on a per document basis by identifying signed documents signed with a specific certificate in the cross domain policy file.
9.0	Support for controlling cross domain access via policy files is introduced. The implementation leverages the Flash model.
8.1.7	An update for 8.x released at the same time as 9.0 enables support for controlling cross domain access via policy files. The implementation leverages the Flash model.

1.6.2.1 allow-access-from-identity usage

`allow-access-from-identity` enables cross domain access on a per document basis by identifying a certified document signed with a specific certificate that should be able to the target domain. Since these documents contain an embedded and unique public key certificate, a SHA-1 hash of the certificate can be used as an identifier, much like a fingerprint. The fingerprint is extracted from

the document and placed in the crossdomain.xml file, thereby providing access. Multiple signatories require multiple `allow-access-from-identity` blocks.

Two types of certificate fingerprints are supported:

- Certificates extracted from a certified document. The signature must be valid. Documents signed with approval (sometimes called "ordinary") signatures are not supported.
- The certificate associated with the digital ID provided to the LiveCycle ES administrator so that the server can Reader enable documents and provide it with additional usage rights.

Tip: The signature must be valid and the certificate must be trusted.

1.6.2.2 Unsupported features

The following is not supported:

- For 9.0 to 9.2 and including the 8.17 update, content-types other than text/x-cross-domain-policy are not supported. Support for all mime types begins with the 9.3 and 8.2 updates.
- Socket connections.
- `allow-http-request-headers-from.`

1.7 Additional resources

More information is available regarding the use of cross-domain policy files. The following resources help describe best practices and provide details for debugging policy file-related issues:

Table 12 Flash resources

Document	Description
Cross-domain policy file usage recommendations for Flash Player	An overview of best practices, including cookie authentication and cross-site forgery requests with respect to your network topology,
Policy File Changes and Usage	A comprehensive look at meta-policies, socket files, and workflows. Written for Flash, but applies to Acrobat.
Setting up a socket policy file server	Details about configuring and using socket policies.

Table 13 Acrobat Resources

Document	Description
Enhanced Security and Privileged Locations	A guide for configuring enhanced security and trusted content on the client.
Cross-domain Access and Configuration	A technical guide client and policy file configuration.
Policy File Changes and Usage	A comprehensive look at meta-policies, socket files, and workflows. Written for Flash, but applies to Acrobat.

2 Index

R

S

T

U