



Ascertia Limited
40 Occam Road
Guildford
Surrey
GU2 7YG
UK
info@ascertia.com

www.ascertia.com

ADSS Auto-File Processor Version 4.0 Admin Manual

Document Version: 4.0.0.1

Document Issued: November 2009

©Copyright Ascertia Ltd, 2009

This document contains commercial-in-confidence material. It must not be disclosed to any third party without the written authority of Ascertia Limited.

Contents

1	Introduction	3
1.1	Scope	3
1.2	Intended Readership	3
1.3	Document Layout	3
1.4	Conventions.....	3
1.5	Technical support	3
1.6	Glossary	4
1.7	References to PKI Standards	4
2	Auto File Processor Concepts & Architecture	5
2.1	Overview.....	5
2.2	Features & Benefits	5
2.3	Deployment Scenarios.....	5
2.4	Multiple Watched Folder Profiles	6
2.5	High Availability / Load balanced Configurations	7
2.6	High Availability Multi-Site Configuration	7
3	System Requirements	8
4	AFP Installation.....	9
4.1	Installing Auto File Processor	9
4.2	Uninstalling the Ascertia AFP Service	9
5	Auto File Processor Configurations.....	10
5.1	AFP Global Settings	10
5.2	AFP Profiles Settings.....	11
5.3	AFP to ADSS Server Communications.....	12
5.4	AFP File Filters	13
5.5	AFP Signature Appearance Settings	13
5.6	AFP Profile based ADSS Server Selection.....	14
5.7	AFP PDF Conversion of Office Documents	14
6	Example Configurations.....	16
6.1	Example 1.....	16
6.2	Example 2.....	17
6.3	Example 3.....	18

1 Introduction

1.1 Scope

This manual describes how to install and configure the Ascertia ADSS Auto File Processor (AFP) software.

1.2 Intended Readership

This manual is intended for Auto File Processor Administrators responsible for its installation and configuration. It is assumed that the reader has a basic knowledge of standard protocols, PKI and IT security.

1.3 Document Layout

This manual is divided into the following chapters:

- Chapter 1: Provides this introduction to the document
- Chapter 2: Explains the Auto File Processor concept and architecture
- Chapter 3: Lists the system requirement for installing Auto File Processor
- Chapter 4: Walks through the installation process
- Chapter 5: Describes various Auto File Processor specific configurations
- Chapter 6: Describes AFP example Configurations

1.4 Conventions

The following typographical conventions are used in this guide to help locate and identify information:

- **Bold text** identifies menu names, menu options, items you can click on the screen, file names, folder names, and keyboard keys.
- `Courier` font identifies code and text that appears on the command line.
- **bold courier** identifies commands that you are required to type in.

1.5 Technical support

If Technical Support is required, Ascertia has a dedicated support team providing debugging assistance, integration assistance and general customer support. Ascertia Support can be accessed in the following ways:

Support Website	www.ascertia.com/support
Support Email	support@ascertia.com
Support MSN Messenger	support@ascertia.com

In addition to the free support service describe above, Ascertia provides formal support agreements with all product sales. Please contact sales@ascertia.com for more details.

A Product Support Questionnaire should be completed to provide Ascertia Support with further information about your system environment. When requesting help it is always important to confirm:

- System Platform details;
- AFP and ADSS Server version numbers and build date;
- Details of the specific issue and the relevant steps taken to reproduce it;
- ADSS Server Database version and patch level;
- The product log files.

1.6 Glossary

ADSS Server	Advanced Digital Signature Services Server. A server-side product from Ascertia for providing signature generation/verification, certificate validation and other trust services.
AFP	Auto-file Processor
CA	Certificate Authority (logical entity responsible for issuing certificates and optionally also CRLs)
CRL	Certificate Revocation List
CMS	Cryptographic Message Syntax (a digital signature format)
DBMS	Database Management System
HSM	Hardware/Host Security Module
HTTP	Hyper Text Transfer Protocol
HTTP/S	HTTP over SSL/TLS connection
Log4j	A robust logging API from Apache
PKCS	Public Key Cryptographic Standards
PKI	Public Key Infrastructure
RSA	Rivest, Shamir, Adleman (public key algorithm)
OCSP	Online Certificate Status Protocol (an IETF protocol for verifying the revocation status of a digital certificate)
SHA	Secure Hash Algorithm (various different algorithms, e.g. SHA-1, SHA-256, SHA-512 etc.)
S/MIME	Secure MIME (standard for signing emails)
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
TA	Trust Authority (authority trusted for issuing certificates, CRLs, OCSP responses and/or time stamps)
TLS	Transport Layer Security (a later version of SSL)
TSA	Time Stamp Authority (authority responsible for issuing timestamp tokens to prove that a document/data existed at a particular time)
TSP	Time Stamp Protocol
XML DigSig	XML Digital Signature standard

1.7 References to PKI Standards

CMS	http://tools.ietf.org/html/rfc3852
PDF Signatures	PDF Public Key Digital Signature and Encryption Specification v3.2 http://www.adobe.com/devnet/pdf/pdf_reference.html
PKCS#7	http://www.fags.org/rfcs/rfc2315.html
PKCS#11	ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf
TSP	http://www.ietf.org/rfc/rfc3161.txt
XML DigSig	http://www.ietf.org/rfc/rfc3275.txt

2 Auto File Processor Concepts & Architecture

2.1 Overview

ADSS Auto File Processor (AFP) is flexible front-end application for the [ADSS Enterprise Server](#). It provides intelligent watched folder automatic monitoring of one or more Windows folders or Unix directories to process batches of documents in an unattended environment.

AFP delivers this by monitoring a set of input folders (sometimes call directories) for documents. Any documents found in a specified input location are processed according to the operator defined AFP policies. Digital signature creation or verification requests are processed by making calls to the ADSS Enterprise Server. The processed documents are delivered to a defined output folder or, if a problem has been detected, an error folder.

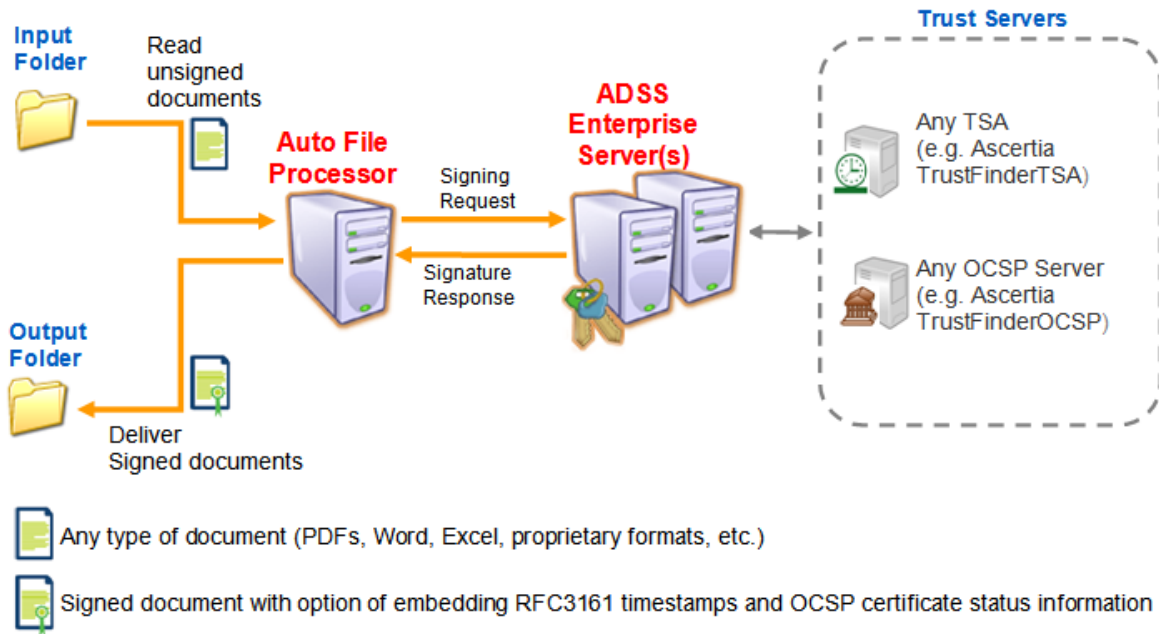
The ADSS Auto File Processor represents one of the simplest ways of utilising the power of the ADSS Enterprise Server within organisations since it requires no programmatic integration.

2.2 Features & Benefits

- **Handles multiple watched folders:**
AFP can manage multiple watched folders profiles, each defines a set of input, output and error folders and the signing policy to be used. It is thus able to handle multiple different business document types.
- **Supports multiple signature formats:**
AFP signing policies can use the full power of ADSS Enterprise Server and thus sign using basic and long-term signatures in a various formats including: PDF, PDF/A, XML DSig, XAdES, PKCS#7, CMS & CAdES. It is thus able to handle multiple business requirements.
- **Supports signing and verification:**
Because AFP delegates the security operations to ADSS Enterprise Server it means that AFP can request documents to be signed or verified. In future archiving will also be offered on the ADSS Enterprise Server and made available in AFP. Thus one AFP system services multiple different processing types.
- **Scalability and Resilience:**
AFP has been designed for high throughput and high availability and it can load balance requests across multiple ADSS Servers to ensure that business systems are unaffected by single point failures.
- **Sophisticated Filtering:**
AFP allows watched folder profiles to include complex rules such as defining sub-folders, filename filters, batch sizes and operational timers. This allows the system to be tuned to optimise throughput and resource utilisation.
- **Security:**
To ensure that only authorised requests are processed AFP authenticates itself to ADSS Enterprise Server using a pre-defined caller ID. For stronger authentication AFP can optionally use Client SSL authentication. ADSS Enterprise Server retains a secure log of all request and response transactions and can provide detailed reports from these.
- **Cost Effective:**
ADSS Enterprise Server is multi-function product and only the modules your business needs need to be licensed. This provides a flexible yet cost-effective solution, with in built investment protection since other modules can be added later to support future business needs. There are no transactional costs - a server can be used to process any number of documents.

2.3 Deployment Scenarios

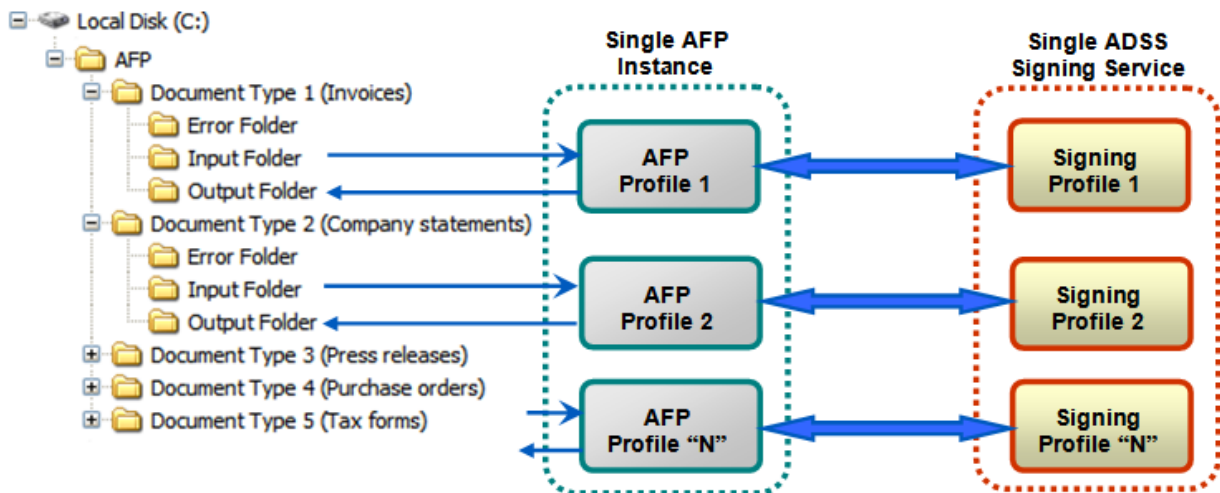
The following diagram shows how AFP can be used with the ADSS Enterprise Server for automated document signing: Back end trust services such as Timestamp Authorities (TSAs) and OCSP Validation Authority servers may be required where long-term signatures are requested:



AFP uses an optimised HTTP communication protocol for communication with the ADSS Enterprise Server to ensure high performance signing operations.

2.4 Multiple Watched Folder Profiles

It is possible to set-up multiple AFP watched folder profiles each with its own set of input, output and error folders as illustrated below. Each AFP watched folder profile can be associated with its own Signing Profile configured on the ADSS Enterprise Server:



The above diagram illustrates multiple AFP profiles processing various document types located in different input folders. The documents are then signed on the ADSS Enterprise Server using unique signing profiles and the resultant file deposited in the appropriate out folder.

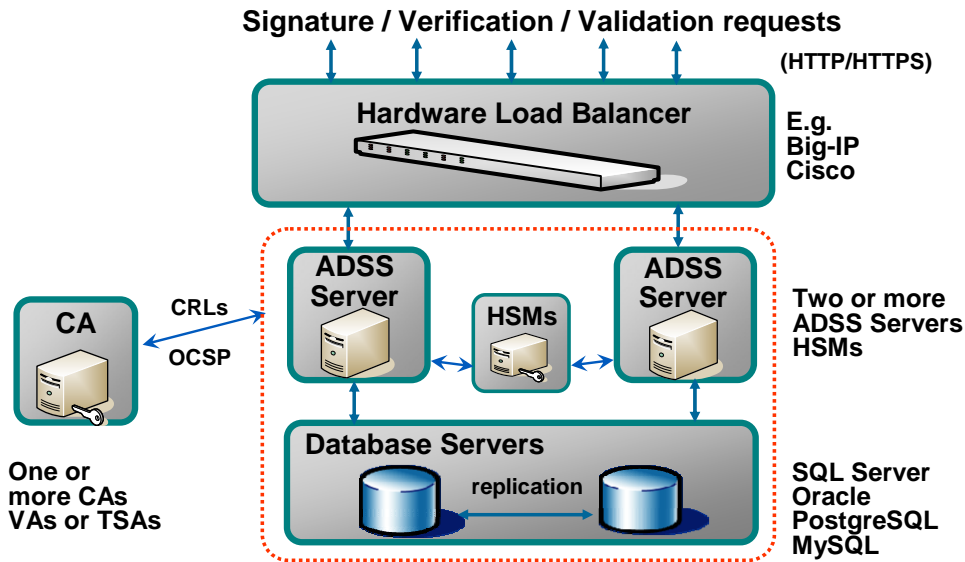
Signing Profiles are configurations on the ADSS Enterprise Server which define the type of signature to produce, including:

- Type of signature to create (PDF, PDF/A, PKCS#7/CMS, XAdES, CAdES etc.)
- Which signing key to use
- Which info to include in the signature (e.g. signing reason, signer's location, signing policy etc.)
- In case of PDF signatures which signature appearance to stamp on the document (e.g. which logos to include, their size and positioning, etc.)

- Plus many additional low-level configurations

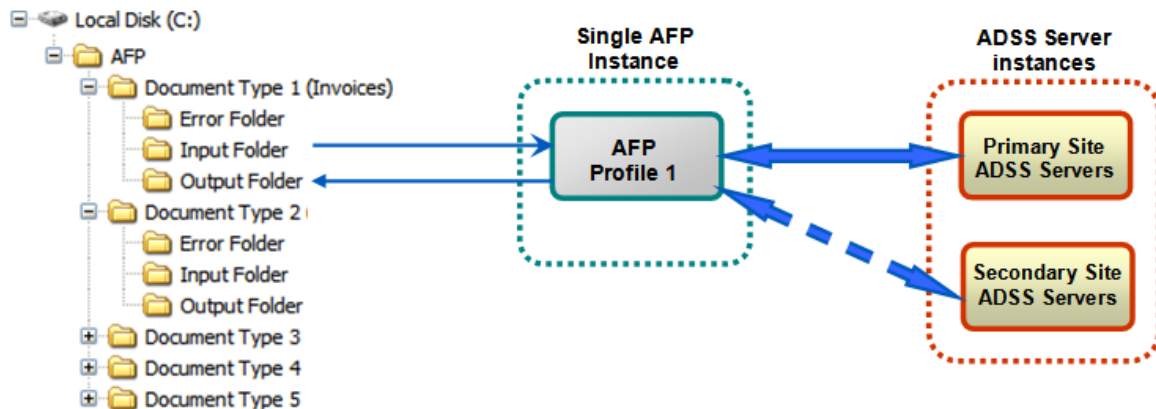
2.5 High Availability / Load balanced Configurations

A load balancer can be deployed to ensure that two or more primary servers are used to provide greater throughput and resilience:



2.6 High Availability Multi-Site Configuration

Multiple ADSS Enterprise Servers can be listed for each AFP profile to ensure if the primary site ADSS Server instances goes down unexpectedly then a back-up site ADSS Server will be automatically invoked:



3 System Requirements

The following table summarizes the minimum requirements for installing Auto-File Processor (AFP):

Component	Minimum Requirements
Operating System	Windows 2003 Server + SP1 Also Windows 2000 Server + SP4 or Solaris 10 on Sparc 32/64 bit hardware Linux Suse 10 and Centos 5 on 32 bit hardware Other Linux systems – ask for availability
CPU/RAM	A modern fast CPU with 1 GB RAM 2GB or more is recommended if larger documents are being processed.

4 AFP Installation

4.1 Installing Auto File Processor

First extract the Auto File Processor (AFP) installation zip to any directory where you want AFP to be installed. The extracted directory will contain following folders:

- bin
- conf
- docs
- jre1.6.0_13
- lib
- logs
- setup

To install AFP run setup.bat located within the <AFP Installation Directory>\setup folder.



To install AFP on the Solaris platform open the setup.sh file within the <AFP Installation Directory>\setup folder, in a text editor. Set the value for the "INSTALL_PATH" variable with the AFP installation directory path e.g. "/export/home/AFP"
Run the command "chmod +x setup.sh" to mark the "setup.sh" file as executable.
Now run the "setup.sh" file using the following command: ./setup.sh

AFP setup is not like other installations that create desktop icons, shortcuts, etc. Instead it is a set of zipped files. Running setup.bat (or setup.sh on Solaris) creates a basic configuration and environment to allow AFP to run successfully. After the installation further configuration work is required as described in the next section.

The Auto File Processor Service is installed as a Windows Service under the name Ascertia-AFP. To run AFP, select the service and click start button. To stop the service, click on stop button.



On Solaris platform use the below commands to start and stop the AFP daemon:
/etc/init.d/wrapperd-AFP start
/etc/init.d/wrapperd-AFP stop



IMPORTANT
The Ascertia-AFP service only reads the afp.xml configuration file when it starts. AFP must therefore be restarted whenever a change is made to the afp.xml configuration file.

4.2 Uninstalling the Ascertia AFP Service

To uninstall Ascertia AFP Service, Run the uninstall_service.bat file located at <AFP Installation Directory>/setup/bin.



To uninstall AFP on Solaris platform, go to the location: <AFP Installation Directory>/setup/bin.
Run the below command to uninstall the AFP daemon:
./uninstall_service.sh

5 Auto File Processor Configurations

This section describes the Auto File Processor specific configurations. All the configurations are created using an XML based configuration file. This `afp.xml` file is located at `<Installation Directory>/conf` folder. The following sections explain each part of this file.

When reviewing these settings it is important to note that some XML elements are required and some are optional. If an element is not required then you must (a) omit the tag, or (b) comment the tag, or (c) leave the value empty. Note: Addresses cannot be left empty.

IMPORTANT: Whenever the `afp.xml` file is changed the Ascertia-AFP service must be restarted to accept the changes.

Example values are shown in **blue font** below to identify them more clearly.

5.1 AFP Global Settings

The Global Settings are:

```
<AFP>
  <Settings>
    <OriginatorId>samples_test_client</OriginatorId>
    <ServiceAddress>
      <PrimaryAddress>http://localhost:8777/adss/signing/hdsi</PrimaryAddress>
    <SecondaryAddress>http://localhost:8777/adss/signing/hdsi</SecondaryAddress>
    </ServiceAddress>
    <ProxySettings>
      <ProxyHost>192.168.0.1</ProxyHost>
      <ProxyPort>8090</ProxyPort>
      <ProxyUserName>proxy-user1</ProxyUserName>
      <ProxyPassword>proxy-password1</ProxyPassword>
    </ProxySettings>
    <ClientAuthPfxSettings>
      <PfxFilePath>client.pfx</PfxFilePath>
      <PfxPassword>pfx-password</PfxPassword>
    </ClientAuthPfxSettings>
  </Settings>
```

XML Tags	Description
AFP	AFP (Auto File Processor) is the root element
SETTINGS	The elements in between <code><Settings></code> and <code></Settings></code> tag are used for global configurations.
ORIGINATOR ID	Defines the client ID for this Auto File Processor and is included in the request messages which are sent to ADSS server. This client ID needs to be registered within the ADSS Client Manager module.
SERVICE ADDRESS	The elements in between <code><ServiceAddress></code> and <code></ServiceAddress></code> tag are used for Defining the ADSS Server primary and secondary addresses. If not required then the primary address OR secondary address tags must either be omitted or commented. An empty address value must not be used.
PRIMARY ADDRESS	Defines the primary ADSS Server address that will be used by default for all profiles unless a profile defines an alternative.
SECONDARY ADDRESS	Defines the secondary ADSS Server address that will be used by default for all profiles unless a profile defines an alternative.

XML Tags	Description
PROXY SETTINGS	The elements in between <ProxySetting> and </ProxySetting> tag are used for defining the proxy settings
PROXY HOST	Defines the IP address or machine name of the proxy server
PROXY PORT	Defines the port no. for communication with the proxy host
PROXY USER NAME	Defines the user name for the proxy user authentication
PROXY PASSWORD	Defines the password for proxy user authentication
CLIENT AUTH PFX SETTINGS	The elements in between <ClientAuthSettings> and </ClientAuthSettings > tag are used for defining the client authentication settings
PFX FILE PATH	Defines the path of the private key to be used for client SSL authentication when communicating with the ADSS Server
PFX PASSWORD	Defines the password for the client SSL authentication private key



If globally defined primary and secondary addresses are not required then comment the element "<ServiceAddress>" instead of commenting the primary and secondary addresses individually. **In this case the ADSS Server address must be configured in each AFP profile.**

5.2 AFP Profiles Settings

The next set of configuration settings deal with file processing. By default afp.xml has 4 sample profiles:

1. PDF Signing - for sending PDF documents to ADSS Server for signing.
2. Empty Signature Field Creation - for creating blank signature fields inside a PDF document
3. PDF File Signing with Local Hashing - for use when you wish the PDF document to be hashed locally and the hash alone sent to the ADSS Server for signing. Once the signed object is returned this is embedded into the PDF document by AFP
4. Conversion – for use when you wish to have other documents converted to PDF

The Operator can create/configure as many profiles as needed. The example below shows all possible values that a profile can have:

```
<Profiles>
  <Profile status="enable" localHash="true" type="signing">
    <InputFolderPath>/export/home/input_folder</InputFolderPath>
    <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
    <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
    <SigningProfile>adss:signing:profile:008</SigningProfile>
    <SigningCertificate>samples_test_signing_certificate</SigningCertificate>
    <HashAlgorithm>SHA1</HashAlgorithm>
  </Profile>
</Profiles>
```

XML Tags	Description
PROFILES	The elements between <Profiles> and </Profiles> are used to define multiple profile configurations.
PROFILE	Defines a profile configuration that can include the following child nodes: <ol style="list-style-type: none"> 1. status: to enable or disable a profile 2. localhash: set to true to enable local hashing 3. Type: to configure either signing or PDF conversion

XML Tags	Description
INPUT FOLDER PATH	Defines the path of the input folder. Files are read and deleted from here
OUTPUT FOLDER PATH	This defines the path of the output folder. Successfully processed files are written here.
ERROR FOLDER PATH	This defines the path of the error folder. Any files that cannot be processed successfully are placed here.
SIGNING PROFILE	Identifies the <u>optional</u> Signing Profile ID to be included in the request messages sent to the ADSS Server. This Signing Profile ID is defined within the ADSS Signing Service module. It is recommended the tag is commented when not required e.g.: <pre><!-- <SigningProfile>adss:signing:profile:001</SigningProfile> --></pre>
SIGNING CERTIFICATE	Identifies the <u>optional</u> Signing Certificate alias to be included in the request messages that are sent to ADSS Server. If used this over-rides the default signing certificate. This signing certificate is defined within the ADSS Key Manager module. It is recommended the tag is commented when not required e.g.: <pre><!-- <SigningCertificate>Invoice Signing Cert</SigningCertificate> --></pre>
HASH ALGORITHM	Identifies the hash algorithm to be used for local hashing purposes only. It is recommended to comment the tag when not required e.g. <pre><!-- <HashAlgorithm>SHA1</HashAlgorithm> --></pre>

5.3 AFP to ADSS Server Communications

The next set of configuration settings deal with AFP to ADSS Server communication:

```
<ConcurrentRequests>10</ConcurrentRequests>
<IdleSleepTime>5</IdleSleepTime>
<ConnectionTimeout>40</ConnectionTimeout>
<FailureRetries>3</FailureRetries>
  <FilePostfix>-processed</FilePostfix>
```

XML Tags	Description
CONCURRENT REQUESTS	Defines the number of files that will be read and sent to ADSS Server in each 'batch' read operation.
IDLE SLEEP TIME	Defines the number of seconds that this AFP profile will wait before the next 'batch' read operation. For example when there are no files to process then the AFP thread can be set to sleep for a set number of seconds. This is timed from when the last batch process completed.
CONNECTION TIMEOUT	Defines the number of seconds that AFP will attempt to connect to a given ADSS Server
FAILURE RETRIES	Defines how many times AFP tries to resend failed requests. A value of 3 is recommended
FILE POST FIX	Defines an optional postfix string to be added to the processed filename. e.g. to add "-s" to a file name use <code><FilePostfix>-s</FilePostfix></code> e.g. to leave the filename unchanged use <code><FilePostfix></FilePostfix></code>

5.4 AFP File Filters

The next set lot of configuration settings deal with file filters:

```
<InputFileFilter>
  <FileExtension>pdf</FileExtension>
</InputFileFilter>
```

XML Tags	Description
INPUT FILTER	Defines Profile filter criteria.
FILE EXTENSION	Defines the type of file that AFP will process based on the file extension.

5.5 AFP Signature Appearance Settings

The next set of configuration settings deal with signature appearance:

```
<PdfSignatureSettings>
  <SignatureField>Signature1</SignatureField>
  <SigningArea>TOP_LEFT</SigningArea>
  <SigningPage>1</SigningPage>
  <SigningReason>I approve this document</SigningReason>
  <SigningLocation>London</SigningLocation>
  <ContactInfo>sales@ascertia.com</ContactInfo>
  <CompanyLogo>/export/home/company_logo.jpg</CompanyLogo>
  <HandSignature>/export/home/hand_signature.jpg</HandSignature>
  <CertifyPermission>CERTIFIED_NO_CHANGES_ALLOWED</CertifyPermission>
  <FontRepository>/export/home/fonts</FontRepository>
  <SignedBy>John Doe</SignedBy>
  <SignerCertificate>/export/home/signer1.cert</SignerCertificate>
  <SignatureAppearance>/export/home/appearance1.xml</SignatureAppearance>
</PdfSignatureSettings>
```

XML Tags	Description
PDF SIGNATURE SETTINGS	The elements between < PdfSignatureSettings > and </ PdfSignatureSettings > can be used to define PDF signature appearance settings that override the settings configured in the signing profile.
SIGNATURE FIELD	Optional: Defines the name of the target (empty) signature field to be signed.
SIGNING AREA	Optional: Defines the location on the PDF document where a signature is to be placed. Possible values are: TOP_LEFT, TOP_RIGHT, CENTER, BOTTOM_LEFT and BOTTOM_RIGHT
SIGNING PAGE	Optional: Defines the page number of the PDF document where signatures are needed to be placed. Permitted values are 1 to the maximum page number of the document being processed.
SIGNING REASON	Optional: Defines the “signing reason” text added within the signature
SIGNING LOCATION	Optional: Defines the “signing location” text added to the signature
CONTACT INFO	Optional: Defines the “contact” text added to the signature
COMPANY LOGO	Optional: Defines the <u>full path</u> of the company logo image to be added

XML Tags	Description
	to the signature
HAND SIGNATURE	Optional: Defines the <u>full path</u> of the hand signature image to be added to the signature
CERTIFY PERMISSION	Optional: Defines the permissions for Certify signing. The permitted values are: (only valid for local hashing) CERTIFIED_NO_CHANGES_ALLOWED CERTIFIED_FORM_FILLING CERTIFIED_FORM_FILLING_AND_ANNOTATIONS
FONT REPOSITORY	Optional: Defines the full path for the font to be used if the font needs to be embedded in the target document (only valid for local hashing)
SIGNED BY	Optional: Defines the text to be inserted in the “Signed By” field of the signature (only valid for local hashing)
SIGNER CERTIFICATE	Optional: Defines the full path of the signing certificate. By default the common name of this certificate would be used in the “Signed By” field of the signature (only valid for local hashing)
SIGNATURE APPEARANCE	Optional: Defines the path of the signature appearance file. The appearance of the signature is controlled by the configurations in the signature appearance file (only valid for local hashing)

5.6 AFP Profile based ADSS Server Selection

It is possible to define profile specific ADSS Server addresses over-riding the AFP global settings.

```
<ServiceAddress>
  <PrimaryAddress>http://localhost:8777/adss/signing/hdsi</PrimaryAddress>
  <SecondaryAddress>http://localhost:8777/adss/signing/hdsi</SecondaryAddress>
</ServiceAddress>
```

XML Tags	Description
SERVICE ADDRESS	The elements in between <ServiceAddress> and </ServiceAddress> tag are used for Defining the ADSS Server primary and secondary addresses. If not required then the primary address OR secondary address tags must either be omitted or commented. An empty address value must not be used.
PRIMARY_ADDRESS	Defines the primary ADSS Server address for this profile.
SECONDARY_ADDRESS	Defines the secondary ADSS Server for this profile. This address is used if the primary address is not available.



If primary and secondary addresses are not required to be defined in a profile (because the globally defined values are to be used) then comment the root element “<ServiceAddress>” instead of commenting the primary and secondary addresses individually.

5.7 AFP PDF Conversion of Office Documents

For conversion of office documents to PDF, the profile type should be set to “conversion” and the input file filter should specify one of the following types “doc”, “xls”, “ppt”, “rtf” and “txt”. AFP uses

OpenOffice® to convert these file types into PDF documents. Use the following command to start the OpenOffice service on the AFP machine or another machine on the same LAN:

```
soffice -headless -accept="socket,host=0.0.0.0,port=8100;urp;" --nofirststartwizard
```

The following configurations should be provided within the document conversion profile in the afp.xml file to identify the instance of OpenOffice:

```
<ConversionMethod name="OpenOffice">
  <Host>localhost</Host>
  <Port>8100</Port>
</ConversionMethod>
```

XML Tags	Description
ConversionMethod	The elements in between <ConversionMethod> and </ConversionMethod> tag are used for defining the conversion application, its host name and the port (currently only OpenOffice is supported).
Host	This element defines the host name or IP of the machine where OpenOffice is running.
Port	This element defines the port on which OpenOffice is running, default is 8100.

6 Example Configurations

This chapter provides some specific examples of AFP configurations. First a text explanation of the scenario is provided and then the complete XML file to execute the scenario described.

6.1 Example 1

This example deals with sending a PDF file for signing to ADSS Server using addresses define in the Profile tag. The Profile also defines the Originator ID, signing key and signing profile data needed by the ADSS Server. Stop the AFP service, make the below changes, place the respective files in the Input folder and now restart the Ascertia-AFP service:

```
<!-- PDF hashing and signing on the ADSS Server -->
  <Profile status="enable" localHash="false" type="signing">
    <InputFolderPath>/export/home/input_folder</InputFolderPath>
    <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
    <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
    <SigningProfile>adss:signing:profile:01 1</SigningProfile>
    <SigningCertificate>samples_test_sgning_certificate</SigningCertificate>
    <ConcurrentRequests>10</ConcurrentRequests>
    <IdleSleepTime>5</IdleSleepTime>
    <ConnectionTimeout>40</ConnectionTimeout>
    <FailureRetries>1</FailureRetries>
    <FilePostfix>-processed</FilePostfix>
    <PdfSignatureSettings>
      <!-- <SignatureField>Signature1</SignatureField> -->
      <SigningArea>TOP_LEFT</SigningArea>
      <SigningPage>15</SigningPage>
      <SigningReason>I approve this document</SigningReason>
      <SigningLocation>London</SigningLocation>
      <ContactInfo>sales@ascertia.com</ContactInfo>
    </PdfSignatureSettings>
    <InputFileFilter>
      <FileExtension>PDF</FileExtension>
    </InputFileFilter>
    <ServiceAddress>
      <PrimaryAddress>http://Machine1:8777/adss/signing/hdsi</PrimaryAddress>
      <SecondaryAddress>http://Machine2:8777/adss/signing/hdsi</SecondaryAddress>
    </ServiceAddress>
  </Profile>
```



Input and output folder paths in the AFP examples are for a Windows platform. Please use the relevant paths when deploying AFP on the Solaris platform.



Multiple profiles for server side signing, hash signing, empty field creation and document conversion can be used together for parallel processing of documents. If a specific profile is not being used then mark this as disabled and keep the rest of the profiles enabled.

Notes:

- Global Settings could have been used to define the ADSS Servers
- The Filename could be changed to something else
- Other signing profiles could have been used
- Additional profiles could be added and could be enabled / disabled.

6.2 Example 2

This is an example of defining multiple profiles sending different type of signing requests to ADSS server using defined ADSS Server address in the profiles, in which Originator ID, signing key and signing profile for File signing is configured within the ADSS Server. Stop the AFP service, make the changes shown below, place the test files in the Input folder and start the Ascertia-AFP service:

```
<AFP>
  <Settings>
    <OriginatorId>samples_test_client</OriginatorId>
    <!-- <ServiceAddress>
      <PrimaryAddress>http://Machine1:8777/adss/signing/hdsi</PrimaryAddress>
      <SecondaryAddress>http://Machine2:8777/adss/signing/hdsi</SecondaryAddress>
    </ServiceAddress> -->
    <ProxySettings>
      <ProxyHost>192.168.0.1</ProxyHost>
      <ProxyPort>8090</ProxyPort>
      <ProxyUserName>user1</ProxyUserName>
      <ProxyPassword>password1</ProxyPassword>
    </ProxySettings>
    <ClientAuthPfxSettings>
      <PfxFilePath>F:\Certs\client.pfx</PfxFilePath>
      <PfxPassword>password2</PfxPassword>
    </ClientAuthPfxSettings>
  </Settings>
  <Profiles>
    <!-- PDF signing at server -->
    <Profile status="enable" localHash="false" type="signing">
      <InputFolderPath>/export/home/input_folder </InputFolderPath>
      <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
      <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
      <SigningProfile>adss:signing:profile:011</SigningProfile>
      <SigningCertificate>samples_test_sgng_certificate</SigningCertificate>
      <ConcurrentRequests>10</ConcurrentRequests>
      <IdleSleepTime>5</IdleSleepTime>
      <ConnectionTimeout>40</ConnectionTimeout>
      <FailureRetries>1</FailureRetries>
      <FilePostfix>-processed</FilePostfix>
      <InputFileFilter>
        <FileExtension>xml</FileExtension>
      </InputFileFilter>
      <ServiceAddress>
        <PrimaryAddress>http://Machine1:8777/adss/signing/hdsi</PrimaryAddress>
        <SecondaryAddress>http://Machine2:8777/adss/signing/hdsi</SecondaryAddress>
      </ServiceAddress>
    </Profile>
    <!-- PDF empty signature field(s) creation -->
    <Profile status="disable" type="signing">
      <InputFolderPath>/export/home/input_folder</InputFolderPath>
      <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
      <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
      <SigningCertificate>samples_test_signing_certificate</SigningCertificate>
      <ConcurrentRequests>10</ConcurrentRequests>
      <IdleSleepTime>30</IdleSleepTime> <!-- seconds -->
      <ConnectionTimeout>40</ConnectionTimeout> <!-- seconds -->
      <FailureRetries>2</FailureRetries>
      <FilePostfix>-processed</FilePostfix>
      <InputFileFilter>
        <FileExtension>pdf</FileExtension>
      </InputFileFilter>
    <PdfEmptySignatureSettings>
      <EmptySignatureProfile>adss:signing:profile:005</EmptySignatureProfile>
    </PdfEmptySignatureSettings>
    <ServiceAddress>
      <PrimaryAddress>http://localhost:8777/adss/signing/hesi</PrimaryAddress>
      <SecondaryAddress>http://localhost:8777/adss/signing/hesi</SecondaryAddress>
    </ServiceAddress>
  </Profiles>
</AFP>
```

```

        </ServiceAddress>
    </Profile>
    <!-- PDF file signing with local hashing -->
    <Profile status="enable" localHash="true" type="signing">
        <InputFolderPath>/export/home/input_folder</InputFolderPath>
        <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
        <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
        <SigningProfile>adss:signing:profile:008</SigningProfile>
        <SigningCertificate>samples_test_signing_certificate</SigningCertificate>
        <HashAlgorithm>SHA1</HashAlgorithm>
        <ConcurrentRequests>10</ConcurrentRequests>
        <IdleSleepTime>30</IdleSleepTime> <!-- seconds -->
        <ConnectionTimeout>40</ConnectionTimeout> <!-- seconds -->
        <FailureRetries>2</FailureRetries>
        <FilePostfix>-processed</FilePostfix>
        <InputFileFilter>
            <FileExtension>pdf</FileExtension>
        </InputFileFilter>
        <PdfSignatureSettings>
            <SignatureField>Signature1</SignatureField>
            <SigningArea>TOP_LEFT</SigningArea>
            <SigningPage>1</SigningPage>
            <SigningReason>I am author of this document</SigningReason>
            <SigningLocation>London</SigningLocation>
            <ContactInfo>support@ascertia.com</ContactInfo>
            <CompanyLogo>/export/home/company_logo.jpg</CompanyLogo>
            <HandSignaturez>/export/home/hand_signature.jpg</HandSignature>
            <CertifyPermission>CERTIFIED_NO_CHANGES_ALLOWED</CertifyPermission>
            <FontRepository>/export/home/fonts</FontRepository>
            <SignedBy>John Doe</SignedBy>
            <SignerCertificate>/export/home/signer1.cert</SignerCertificate>
            <SignatureAppearance>/export/home/appearance1.xml</SignatureAppearance>
        </PdfSignatureSettings>
        <ServiceAddress>
            <PrimaryAddress>http://Machine3:8777/adss/signing/hdsi</PrimaryAddress>
            <SecondaryAddress>http://Machine4:8777/adss/signing/hdsi</SecondaryAddress>
        </ServiceAddress>
    </Profile>
</Profiles>
</AFP>

```

6.3 Example 3

This example deals with processing a word document file for conversion from OpenOffice. Stop the AFP service, make the changes shown below, place the test files in the Input folder and restart the Ascertia-AFP service:

```

    <Profile status="enable" type="conversion">
        <InputFolderPath>/export/home/input_folder</InputFolderPath>
        <OutputFolderPath>/export/home/output_folder</OutputFolderPath>
        <ErrorFolderPath>/export/home/error_folder</ErrorFolderPath>
        <ConcurrentRequests>1</ConcurrentRequests>
        <IdleSleepTime>30</IdleSleepTime> <!-- seconds -->
        <ConnectionTimeout>40</ConnectionTimeout> <!-- seconds -->
        <FailureRetries>2</FailureRetries>
        <FilePostfix>-processed</FilePostfix>
        <InputFileFilter>
            <FileExtension>doc</FileExtension>
        </InputFileFilter>
        <ConversionMethod name="OpenOffice">
            <Host>localhost</Host>
            <Port>8100</Port>
        </ConversionMethod>
    </Profile>

```

*** End of document ***