



Ascertia Limited
40 Occam Road
Surrey Research Park
Guildford
Surrey
GU2 7YG

Tel: +44 1483 685500
Fax: +44 1483 573704

www.ascertia.com

Secure Email Server, 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.

Commercial-in-Confidence

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	5
2	Secure Email Server Concepts & Architecture	6
2.1	Introduction.....	6
2.2	Secure Email Server Architecture.....	6
2.3	ADSS Server	8
2.4	Outgoing Email & Attachment Signing Process	8
2.5	Verification of Incoming Signed Emails & Attachments	9
2.6	Key Management	10
2.7	Security.....	11
3	System Requirements	12
4	SES Installation.....	13
4.1	Installing Secure Email Server.....	13
4.2	Uninstalling Ascertia SES Service	13
5	Apache James Configurations	14
5.1	DNS Configuration.....	14
5.2	POP3 Server	14
5.3	SMTP Server.....	14
5.4	Server Wide Configuration.....	15
5.5	Authenticated SMTP (SMTP AUTH).....	15
5.6	Remote Manager	16
5.7	Setting Up administrator Account	16
6	Secure Email Server Configurations.....	18
7	Example Configurations.....	25
7.1	Example-1	25
7.2	Example-2	26
7.3	Example-3	27
7.4	Example-4	30
7.5	Example-5	33
7.6	Example-6	35
7.7	Example-7	37
7.8	Example-8	40
7.9	Example-9	41

1 Introduction

1.1 Scope

This manual describes how to install and configure the Ascertia Secure Email Server (SES).

1.2 Intended Readership

This manual is intended for Secure Email Server Administrators responsible for its installation and configuration. It is assumed that the reader has a basic knowledge of standard email 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 Secure Email Server concept and architecture
- Chapter 3: Lists the system requirement for installing Secure Email Server
- Chapter 4: Walks through the installation process
- Chapter 5: Describes general Apache James Mail Server configurations
- Chapter 6: Describes various Secure Email Server specific configurations
- Chapter 7: Describes SES 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;
- SES 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	Advanced Digital Signature Services (a server-side product from Ascertia for providing signature generation/verification, certificate validation and other trust services)
Apache James	Open Source MTA Server (available from http://james.apache.org/)
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
IMAP	Internet Message Access Protocol
Log4j	A robust logging API from Apache
MTA	Mail Transfer Agent
PKCS	Public Key Cryptographic Standards
PKI	Public Key Infrastructure
POP3	Post Office Protocol version 3
RSA	Rivest, Shamir, Adleman (public key algorithm)
OCSP	Online Certificate Status Protocol (an IETF protocol for verifying the revocation status of a digital certificate)
SES	Secure Email Server (an enhanced Apache James mail server from Ascertia which enables e-Trust services for outgoing and incoming emails and attachments).
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
S/MIME	http://www.ietf.org/rfc/rfc3851.txt
TSP	http://www.ietf.org/rfc/rfc3161.txt
XML DigSig	http://www.ietf.org/rfc/rfc3275.txt

2 Secure Email Server Concepts & Architecture

2.1 Introduction

Today email is one of the most popular applications on the Internet. This position has been earned because of its clear business benefits. However for several years, there has been a dramatic rise in email fraud and phishing, so much so that the "From:" field in an email message or the message content itself can no longer be trusted as a face value. It is easy to spoof these details, opening the way for attackers to create and use false identities with minimum effort. The recent record level fraud committed at a leading European bank, which at the core level involved email spoofing, serves as clear evidence of what happens when emails are trusted blindly without any security trails.

To overcome such threats in electronic communication, Ascertia provides the multi-purpose Advanced Digital Signature Services (ADSS) Server as a solution for digitally signing and verifying any type of electronic document so that it's content can be easily trusted and the identity of its authors, reviewers and/or approvers can be automatically verified.

Ascertia offers the Secure Email Server as a client application to the ADSS Server with the objective of managing the:

- Signing of outgoing emails (using standard S/MIME digital signatures which are verifiable by most email clients like Microsoft Outlook, Lotus Notes and Thunderbird).
- Signing of outgoing email attachments (using PDF digital signatures, XML Digital signatures or PKCS#7/CMS signatures. Advanced long-term signature profiles using CAAdES and XAdES as well as the PDF equivalent are also supported).
- Verification of incoming signed emails (including sender's identity check using real-time OCSP)
- Verification of incoming signed email attachments (including sender's identity check using real-time OCSP).

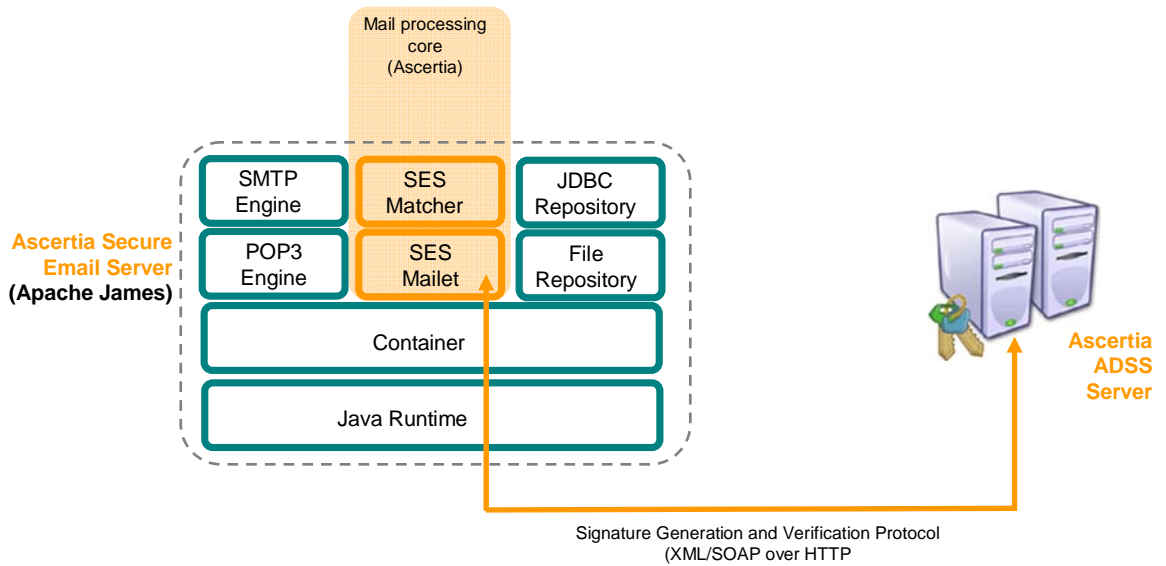
The following sections explain the SES architecture and how the SES and ADSS Server collaborate to achieve the above objectives.

2.2 Secure Email Server Architecture

Ascertia Secure Email Server is drop-in MTA server which supports both SMTP and POP3 protocols. It is built using the open source Apache James, a popular platform-independent pure java mail server. Apache James provides a mail application platform with an embedded standard extension mechanism built around:

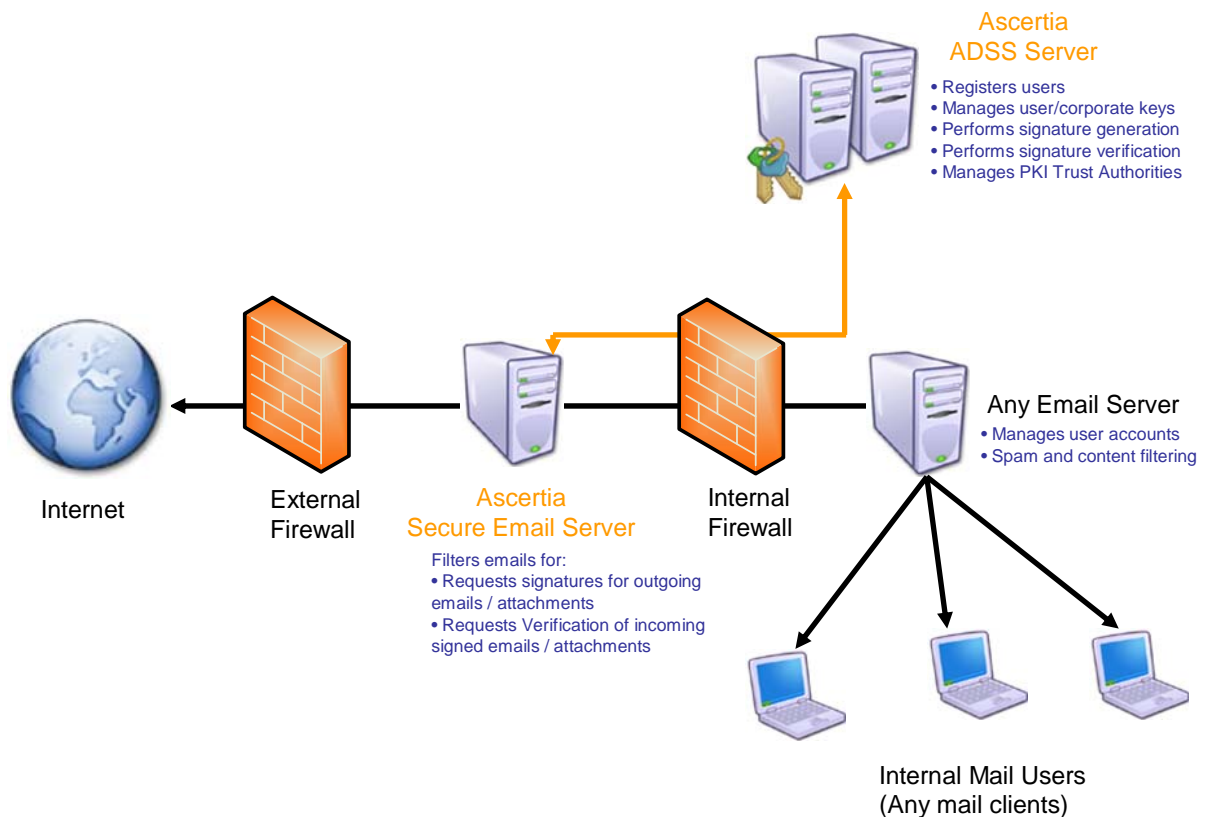
- Matchers: Message selection code, i.e. specific code written to filter emails which are then passed to the maillet.
- Maillets: Message processing code, i.e. specific code written which then processes the filter email, e.g. in case of SES the maillet calls ADSS Server to digitally sign or verify the email (or email attachment). Therefore the Apache James server can be seen as a container for the matcher and maillet.

The high-level architecture of Secure Email Server is shown below. The green boxes show standard Apache James modules including the service engines SMTP and POP3 which deliver email to Apache James. The core mail processing engine shown with the orange boxes represents Ascertia specific matcher and maillet for digital signature creation and verification functionality:



Note in future Ascertia plans to provide extended functionality which allows email archiving and email encryption/decryption services through this matcher/mailet architecture. For further details on these services, please contact info@ascertia.com.

A typical deployment of Secure Email Server is shown below, with an internal email server handling emails as normal, but then passing these to the Secure Email Server for signing. If the SES Matcher determines that the email requires signing the SES Maillet then sends a signing request message to the ADSS Server (operating on secure network behind an internal firewall)



Similarly for incoming emails, the SES Matcher can filter emails and attachments which are signed and then the SES Maillet can make a request to the ADSS Server to verify these signatures. The original email and the

verification results are then passed to the internal email server for later delivery to the internal mail recipient(s) or administrators in case of errors in verification.

2.3 ADSS Server

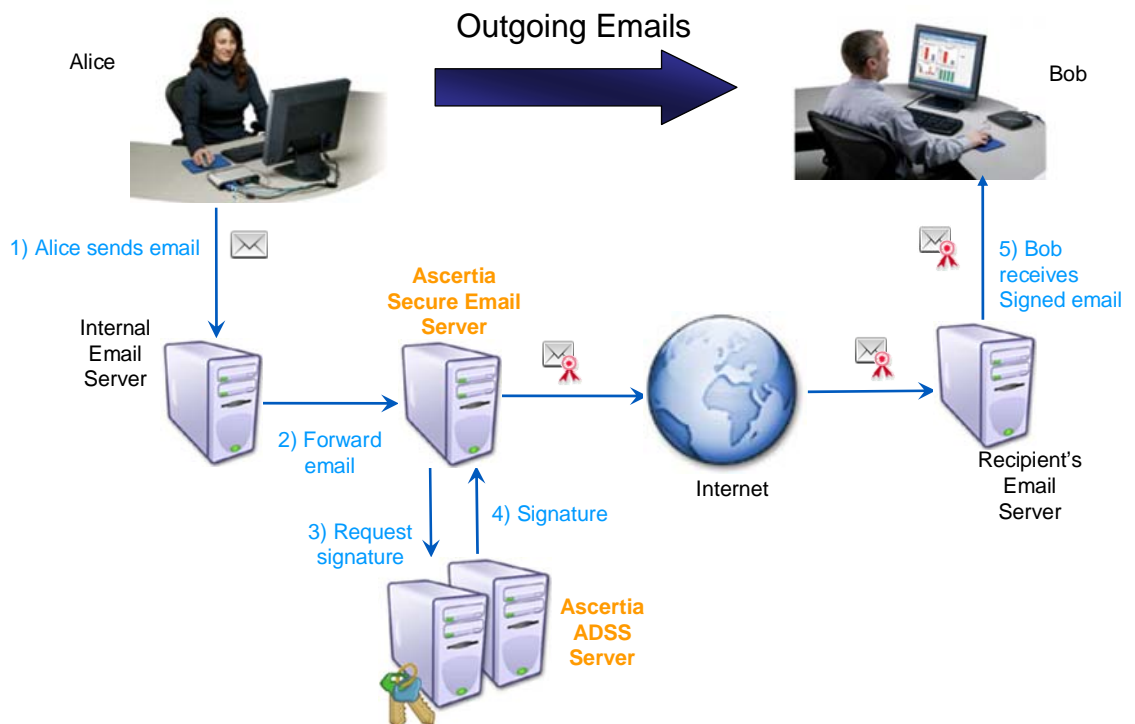
Ascertia ADSS Server is a multi-function application offering the following underlying e-Trust services:

- **Signing Service:** ability to digital sign on the server using server-held keys. It supports most popular signature formats including S/MIME, PDF, XML DSig, PKCS#7, CMS, CAdES-T, CAdES-X-Long, XAdES-T and XAdES-X-Long. The signing service is available through an XML/SOAP web service interface and watched folders.
- **Verification Service:** ability to fully verify the above signature formats, including the complete validation of associated certificate chains. The verification service is available through an XML/SOAP web service interface. Historical signature verification using archived information is also supported.
- **Key Manager & Certification Service:** ability to generate cryptographic keys and certify the public keys via an internal CA or external online or offline CAs. Supports communication with HSMs. This service is also available through an XML/SOAP web services interface.
- **Timestamping Service:** provides cryptographic timestamping service. Required to form long-term signatures with embedding trusted timestamp information.
- **OCSP Validation Authority Service:** provides real-time certificate status information, which can again be embedded in advanced signature formats so that such signatures can be verified in the long term.

ADSS Server can support multiple CAs and multiple PKI trust models, allowing a single centralized trust service to be established. For further details see the ADSS Server Admin Manual.

2.4 Outgoing Email & Attachment Signing Process

The following diagram illustrates a typical scenario where outgoing emails are automatically signed before being released from the company using Secure Email Server:



1. Here the client e.g. Alice sends an email. Her email is sent to her corporate internal email server e.g. her existing email server so that no re-configuration of email clients is necessary.

2. The internal email server then forwards the emails to the Ascertia Secure Email Server. The SES Matcher processes the email to determine if it requires a signature. Typical matcher might filter emails based on:
 - a. Keywords in the From, To, CC, BCC, Subjects and/or email body
 - b. Whether the email contains a particular type of attachment (e.g. PDF, XML, DOC, TXT etc)
 - c. Whether the email is signed
3. If the email matches a particular filter, the Secure Email Server will then make a request to the ADSS Server to perform the signature. Note a number of different operations could be asked from the Secure Email Server, the following options are allowed in the configuration file as explained later:
 - a. SIGN_ATTACHMENT
 - b. SIGN_EMAIL
 - c. SIGN_ATTACHMENT, SIGN_EMAIL (signs both the email and the attachment)
 - d. SIGN_EMAIL, ARCHIVE_EMAIL
 - e. SIGN_EMAIL, ENCRYPT_EMAIL
 - f. SIGN_ATTACHMENT, SIGN_EMAIL, ARCHIVE_EMAIL
 - g. SIGN_ATTACHMENT, SIGN_EMAIL, ENCRYPT_EMAIL
 - h. SIGN_ATTACHMENT, ARCHIVE_EMAIL
 - i. SIGN_ATTACHMENT, ENCRYPT_EMAIL

Note archiving encrypting and decrypting functionality will be added in a future release of the Secure Email Server.
4. ADSS Server on receiving the request processes it and returns the signature back to the Secure Email Server, which embeds the signature into the email and forwards this to the recipients Email Server.
5. The recipient receives the signed email or signed document

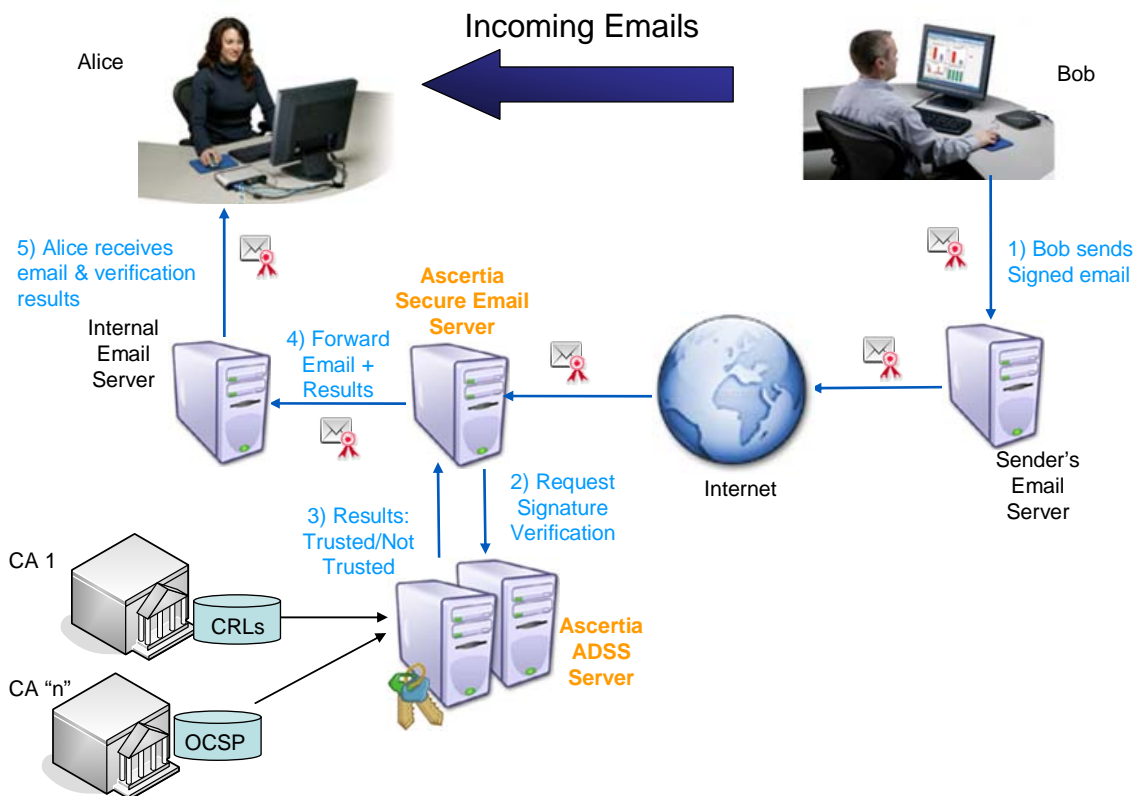
When signing a PDF email attachment (e.g. invoices) it is possible to configure the Secure Email Server Maillet to specify how it wishes the signature appearance to look and where to stamp it on the PDF document. Example signature appearance is shown below:



Normally the Maillet will not want to provide this information in every signing request message to the ADSS Server, so signature profiles can be pre-configured on the ADSS Server and referred to by the Maillet in the request message. Note the language for the above labels is also configurable; see the ADSS Manual for further details.

2.5 Verification of Incoming Signed Emails & Attachments

Similar to the above scenario it is possible for incoming emails which are signed, or where the email attachment is signed, it is possible for the Secure Email Server to automatically verify the signatures and then provide these results to the internal recipient or administrator as depicted below:



1. An external user "Bob" sends a signed email or signed attachment to "Alice" an internal user.
2. When the email enters the corporate network it is routed to the Secure Email Server, where the Matcher filters emails to detect if they contain signatures or if the attachments are signed. The filter as explained above can also be set-up to look for specific keywords in various email fields. If the email or attachment is signed then the Maillet processes it by making a signature verification request to the ADSS Server.
3. The ADSS Server is responsible for verifying digital signatures. It gathers certificate status information from the configured Certificate Authorities (CAs) using either offline CRLs or online OCSP mechanism. The ADSS Server provides a response back to the Maillet on the signature verification results which can be that the signature is "trusted", "not trusted" or "indeterminate" (i.e. the status of the signature cannot be determined because of insufficient information). The Maillet on receiving the response from the ADSS Server can perform various actions depending on the configuration, e.g.:
 - a. If results show signature is trusted, then pass the signed email to the original recipient with the results shown in the email body (or as an attachment)
 - b. If results are not trusted, then pass the signed email to a system administrator.
4. The Secure Email Server then passes the email to the internal corporate email server.
5. This internal email server then delivers the email to the internal user.

2.6 Key Management

One of the benefits of the Secure Email Server and ADSS Server solution is that no cryptographic keys need to be deployed to end-users. End-users in fact don't even need to learn how to sign their emails as the signatures are automatically applied. The signing keys are managed by the ADSS Server in either a secure HSM or held in encrypted form in the ADSS Server database. The choice of how many signing keys to use is optional, e.g.:

- Set-up an organization, departmental, or role level key in the ADSS Server, and sign all emails using this group key. E.g. signing of emails by the Accounts Department, or signing of corporate material using a corporate signing key.
- Set-up unique user keys for each email user. Then the ADSS Server applies the signature using the signing key of the email sender (using the information from the email's "FROM" field to identify the user).

2.7 Security

The solution has been designed for a high-level of practical security. In particular, the secure ADSS Server is used for managing all cryptographic keys and signing/verification operations and other auxiliary trust services. Being an e-Trust service provider, the ADSS Server has been designed with security in mind:

- All service request/responses to the ADSS Server are logged in secure cryptographically-protected logs.
- Operator console access to the ADSS Server is protected using HTTPS with client authentication.
- ADSS server comes with optional dual control facility so that at least two operators are required to make configuration changes and then approve these changes before settings can be changed.
- Client applications like the Secure Email Server need to be registered on ADSS Server, plus the service request messages can optionally be signed by the Secure Email Server.
- The ADSS Server can use any PKCS#11 tamper-resistant HSM for cryptographic processing. Safenet (Luna and Eracom devices) and nCipher HSMs have been tested, others can be supported on request.
- The ADSS Server operates on a secure internal network, e.g. in a secure computing room.
- High-availability is offered with the ability for the Secure Email Server to contact a fall-back ADSS Server instance in case primary server becomes unavailable.

3 System Requirements

The following table summarizes the minimum requirements for installing Secure Email Server (SES):

Component	Requirements
Operating System	Windows 2000 Server + SP4 or Windows 2003 Server + SP1
CPU/RAM	A modern fast CPU with 1 GB RAM

4 SES Installation

4.1 Installing Secure Email Server

First extract the Secure Email Server Installation zip to any directory where you want Secure Email Server to be installed. The extracted directory will contain following folders:

1. conf
2. james-2.3.1
3. jre1.6.0_13
4. logs
5. setup
6. temp
7. docs

To install Secure Email Server on your machine, run **setup.bat** file located inside **<SES Installation Directory>/setup** folder. Launching **setup.bat** will install Ascertia SES Service.



Note the Secure Email Server setup is not like other installations which create desktop icons, shortcuts, asks for target folder where the product is to be installed, etc. Instead it is provided as a zipped set of files and running **setup.bat** merely performs the basic configuration before it can be used. After installation you will still need to perform further configuration as described in the next section.

The Secure Email Service will be installed in <Windows Control Panel/Administrative Tools/Services> under the name Ascertia-SES. To run Secure Email Server, select the service and click **start** button. To stop the service, click on **stop** button.



It is very important to note that restarting of the Ascertia-SES service is essential whenever any change is made in **SES.xml** configuration file.

Note that SES Service can also be run by clicking on **run.bat** file located at: **<SES Installation Directory>/james-2.3.1/bin**.

4.2 Uninstalling Ascertia SES Service

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

5 Apache James Configurations

This section describes Apache James configurations (e.g. add user, configure Domain etc.) to configure mail server. This section only describes the information relevant to using Secure Email Server for signing and verifying emails and not general operation on how to set-up other email services (e.g. POP3 accounts). For this we recommend detailed information available from <http://james.apache.org/index.html>.



Note most of the default configurations are already made in the setup, if you like to change the default configurations, please follow the steps mentioned in this section.

5.1 DNS Configuration

DNS Transport services are controlled by **config.xml** configuration file located inside **<SES Installation Directory>/james-2.3.1/apps/james/SAR-INF** folder. This configuration file affects SMTP remote delivery.

Domain Name System (DNS) is the name resolution protocol for TCP/IP networks, such as the Internet. Client computers query a DNS server to resolve memorable, alphanumeric DNS names to the IP addresses that computers use to communicate with each other. The **dnserver** tag defines the boundaries of this configuration block. It encloses the entire relevant configuration for the DNS server. The behavior of the DNS service is controlled by the attributes and children of this tag.

The standard children of the **dnserver** tag are:

servers - This is a list of DNS Servers to be used by James and are specified by one, or more server elements, which are child elements. Each server element is the IP address of a single DNS server.

```
<servers>
  <server> 127.0.0.1</server>
  <server>166.181.194.205</server>
</servers>
```

5.2 POP3 Server

The Post Office Protocol version 3 (POP3) is an application-layer Internet standard protocol, to retrieve e-mail from a remote server over a TCP/IP connection. The POP3 service is controlled by a configuration block in the **config.xml**. The **pop3server** tag defines the boundaries of this configuration block. It encloses the entire relevant configuration for the POP3 server. The behavior of the POP service is controlled by the attributes and children of this tag.

This tag has an optional boolean attribute - **enabled** - that defines whether the service is active or not. The value defaults to "true" if not present.

```
<pop3server enabled="true">
  <port>110</port>
  <!-- Uncomment this if you want to bind to a specific inetaddress -->
  <bind>192.168.0.181</bind>
  <!--Uncomment this if you want to use TLS (SSL) on this port -->

  <useTLS>true</useTLS>
</pop3server>
```

5.3 SMTP Server

SMTP (Simple Mail Transfer Protocol) is a TCP/IP protocol used in sending and receiving e-mail. However, since it is limited in its ability to queue messages at the receiving end, it is usually used with one of two other protocols, POP3 or IMAP, that let the user save messages in a server mailbox and download them periodically from the server. In other words, users typically use a program that uses SMTP for sending e-mail and either

POP3 or IMAP for receiving e-mail. The SMTP service is controlled by a configuration block in the **config.xml**. The **smtpserver** tag defines the boundaries of this configuration block. It encloses the entire relevant configuration for the SMTP server. The behavior of the SMTP service is controlled by the attributes and children of this tag.

This tag has an optional boolean attribute - **enabled** - that defines whether the service is active or not. The value defaults to "true" if not present.

```
<smtpserver enabled="true">
<!-- port 25 is the well-known/IANA registered port for SMTP -->
  <port>25</port>

<!-- Uncomment this if you want to bind to a specific ip address otherwise it will listen on all interfaces installed
n the system-->

  <bind> 192.168.0.181</bind>

<!-- Uncomment this if you want to use TLS (SSL) on this port -->
<!--
  <useTLS>true</useTLS>

</smtpserver>
```

5.4 Server Wide Configuration

POSTMASTER ADDRESS = the body of this element is the address that the server will consider its postmaster address. This address will be listed as the sender address of all error messages that originate from Apache James.

```
<postmaster>administrator@testdomain.com</postmaster>
```

USERNAMES = this element has no body, but instead has three required boolean attributes. These are **ignoreCase**, **enabledAliases**, and **enableForwarding**. The first of these determines whether email user names will be treated as case-insensitive or not. The second attribute configures whether local user aliasing will be enabled. Finally, the value of the third attribute determines whether forwarding to potentially remote users will be enabled.

```
<usernames ignoreCase="true" enableAliases="true" enableForwarding="true"/>
```

SERVERNAMES - this element determines exactly which mail domains and IP addresses the server will treat as local.

```
<servername>localhost</servername>
  <servername>testmail.com</servername>
  <servername>testmail1.com</servername>
</servernames>
```

5.5 Authenticated SMTP (SMTP AUTH)

Authenticated SMTP is a method of securing your SMTP server. With SMTP AUTH enabled senders who wish to relay mail through the SMTP server (that is, send mail that is eventually to be delivered to another SMTP server) must authenticate themselves to James before sending their message. Mail that is to be delivered locally does not require authentication. This method ensures that spammers cannot use your SMTP server to send unauthorized mail, while still enabling users who may not have fixed IP addresses to send their messages.

Configuring James for Authentication SMTP is a multi-step process. It requires several adjustments of the config.xml. To enable SMTP AUTH, do the following:

Firstly, as mentioned above, SMTP AUTH requires that James be able to distinguish between mail intended for local delivery and mail intended for remote delivery. James makes this determination by matching the domain

to which the mail was sent against the <servername> element of the James configuration block. Any local domains should be explicitly listed as <servername> elements in this section.

```
<servername>localhost</servername>

    <servername>testmail.com</servername>
    <servername>testmail1.com</servername>
</servernames>
```

Secondly, James is configured out of the box so as to not serve as an open relay for spammers. This is done by restricting the IP addresses from which mail will be accepted using the **RemoteAddrNotInNetwork** mailet. This restriction must be lifted before users can send from arbitrary clients. To do this, comment out or remove the mailet tag containing the class attribute "**RemoteAddrNotInNetwork**". This tag can be found in the spoolmanager configuration block, in the root processor configuration.

Comment out this tag.

```
<mailet match="RemoteAddrNotInNetwork=127.0.0.1" class="ToProcessor">
```

Thirdly, set the authRequired element of the **smtpserver configuration block** to "true".

```
<authRequired>true</authRequired>
```

Fourthly, if you wish to ensure that authenticated users can only send email from their own account, you may optionally set the verifyIdentity element of the smtpserver configuration block to "true".

```
<verifyIdentity>true</verifyIdentity>
```

5.6 Remote Manager

In James, user accounts are created through the RemoteManager. So, after installation is complete, the first step to adding users is to configure the RemoteManager

```
<remotemanager enabled="true">
<!--port that will be used to telnet for creating users -->
<port>4555</port>
<!-- Uncomment this if you want to bind to a specific inetaddress -->
<!--
    <bind>192.168.0.181 </bind>
```

5.7 Setting Up administrator Account

You also need to setup administrator accounts so that administrator can login and create users. This is done in the same RemoteManager tag

```
<administrator_accounts>
<!-- CHECKME! -->
    <!-- Change the default login/password. -->
    <account login="root" password="root"/>
</administrator_accounts>
```

After you've done this, restart James to ensure that any changes you've made in the configuration are incorporated into the running system. You are now ready to create user accounts.

Telnet to the host and port on which the RemoteManager is listening. For command-line telnet clients this is generally done by typing "telnet <host> <pass>" where <host> is the James hostname and <port> is the RemoteManager port specified in the James config.xml.

Step-1: Open Command Prompt

And run command, type:

telnet localhost 4555

Step-2: You will be prompted for your administrator userid and password. Enter the values you specified in the James config.xml.

Step-3: After logging in, type "adduser <user> <password>" where <user> is the user name and <password> is the password of the account you wish to create.

E.g. Adduser user1 password123

Please note that the user name should NOT be a complete email address. Rather, all email addresses of the form <user>@<domain> (where <domain> is any of the values specified in the <servername> block) will be delivered to this account by default. Maillet configuration can change this default behavior.

Repeat step 3 for all user accounts you wish to create.

That's it. Your user accounts are now created and can be used by all James services.

6 Secure Email Server Configurations

This section describes the Secure Email Server specific configurations. All the configurations and workflows specific settings for incoming and outgoing emails are created using an XML based configuration file. This **ses.xml** file is located at **<SES Installation Directory>/conf** folder. The following sections explain each part of this file.

```
<SES>
  <Settings>
    <DomainName>companyABC.com</DomainName>
    <AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
    <OriginatorId>ADSS_QAS</OriginatorId>
```

XML Tags	Description
SES	SES (Secure Email server) is the root element
SETTINGS	The tags in between <Settings> and </Settings> tag are used for global configurations.
DOMAIN_NAME	This defines the name of your internal email domain.
ADMIN_EMAIL_ADDRESS	This defines the email address of the Secure Email Server administrator.
ORIGINATOR_ID	This is the client ID of this Secure Email Server 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.

```
<SigningServiceSettings>
  <SigningServiceAddress>
    <PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
    <SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
  </SigningServiceAddress>
  <NoFilterAction>BLOCK_EMAIL</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
  <VerificationServiceAddress>
    <PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
    <SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
  </VerificationServiceAddress>
  <NoFilterAction>BLOCK_EMAIL</NoFilterAction>
  <ResponseNotTrustedAction>BLOCK_EMAIL</ResponseNotTrustedAction>
</VerificationServiceSettings>
```



XML Tags	Description
PRIMARY_ADDRESS	This defines the primary addresses for the ADSS Server Signing and Verification Service.
SECONDARY_ADDRESS	This defines the corresponding secondary addresses for the ADSS Server Signing and Verification Service. This address is used by the Maillet if the primary address is not available.
NO_FILTER_ACTION	This defines what action to take if the maillet is making calls to ADSS Server (for signing and verification) or to a TSA fails to communicate with ADSS Server. BLOCK_EMAIL EMAIL_TO_ADMINISTRATOR EMAIL_TO_RECIPIENTS EMAIL_TO_ADMINISTRATOR_AND_RECIPIENT
RESPONSE_NOT_TRUSTED	This defines what action to take if verification response returned from ADSS Server verification service is not trusted. It contains one of the following rules: BLOCK_EMAIL EMAIL_TO_ADMINISTRATOR EMAIL_TO_RECIPIENTS EMAIL_TO_ADMINISTRATOR_AND_RECIPIENT

```

<ProxySettings>
  <ProxyHost>192.168.1.124</ProxyHost>
  <ProxyPort>1555</ProxyPort>
  <ProxyUserName>root</ProxyUserName>
  <ProxyPassword>admin</ProxyPassword>
  <ProxyDigestAuth>FALSE</ProxyDigestAuth>
</ProxySettings>
<EmailErrorAction>BLOCK_EMAIL</EmailErrorAction>
<ClientAuthPfxSettings>
  <PfxFilePath>d://pfx/path/test.pfx</PfxFilePath>
  <PfxPassword>password</PfxPassword>
</ClientAuthPfxSettings>
</Settings>
    
```

XML Tags	Description
PROXY_HOST	This is the IP address of the proxy server in case communication between the Secure Email Server and ADSS Server needs to go through a proxy machine.
PROXY_PORT	This defines the port number on which proxy host is running.
PROXY_USERNAME	This is the username used to authenticate on the proxy server.
PROXY_PASSWORD	This is the password of the proxy user.
PROXY_DIGEST_AUTHENTICATION	This tag is used to enable/disable the use of digest authentication when the communication between ADSS Server



XML Tags	Description
	and Secure Email Server is through a proxy machine.
EMAIL_ERROR_ACTION	The Email Error Action element defines what to do with the email when an error occurs. It is useful in trapping such emails which failed to reach the recipient due to wrong filter configuration. It contains one of the following values: BLOCK_EMAIL EMAIL_TO_ADMINISTRATOR EMAIL_TO_RECIPIENTS EMAIL_TO_ADMINISTRATOR_AND_RECIPIENT
PFX_FILE_PATH	This is the address where the PFX file is located. The PFX file contains the signing key used by the Secure Email Signer to digitally sign the request messages sent to the ADSS Server.
PFX_PASSWORD	This is the password of the PFX file.

```

<Outgoing>
<Sign>
<Workflow>
<Filter>
<!-- The emails will be filtered based on OR of all the fields under this Element i.e. an email would be filtered
even if a single criterion is met. Secure Email Server does not support mix of and/or filtering. For AND or OR
filter same elements are supported and elements can be omitted. -->
<Or>
  <From contains="user1@companyABC.com" />
  <To contains="*@*.com" />
  <CC contains="*@*.com" />
  <Subject contains="sign" />
  <Body contains="confidential" />
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</Or>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:001">
<!--
It is the name of the signing profile configured in ADSS Server Signing Service to sign an attachment.
-- >
<OverrideableAttributes>
<!-- -
PDF signature appearance elements which are being overridden from the default values defined in signing
profile. -- >
<Attribute>
<Name>SIGNING_LOCATION</Name>
<Value>Surrey, UK</Value>
</Attribute>

<Attribute>
<Name>SIGNING_REASON</Name>
    
```

```

<Value>I certify this document</Value>
</Attribute>

<Attribute>
<Name>CONTACT_INFO</Name>
<Value>111-222-333</Value>
</Attribute>

<Attribute>
<Name>SIGNING_FIELD</Name>
<Value>Signaturefield1</Value>
<!--existing blank signature field name in signing profile. This attribute is useful in signing documents having
blank signature field in them. -->
</Attribute>

<Attribute>
<Name>SIGNING_PAGE</Name>
<Value>1</Value>
<!-- Defines on which page number to sign in the document -->
</Attribute>

<Attribute>
<Name>SIGNING_AREA</Name>
<Value>1</Value>
<!--Possible values are: 1 for Signing at TopLeft location, 2 for Signing at TopRight location, 3 for Signing at
Center location, 4 for Signing at BottomLeft location, 5 for Signing at Bottom Right location-->
</Attribute>

<Attribute>
<Name>VISIBILITY</Name>
<Value>>true</Value>
<!--This attribute has two possible values i.e. true, false. It is used for visibility of signature on the document.
Attribute value set as true will show the visible signature on the document.-->
</Attribute>

</OverrideableAttributes>
</SignAttachment>

<SignEmail profileId="adss:signing:profile:002" />
</ProfileSelection>

<!-- Here you define the signing certificate aliases and password for the signing. For signing with a default
key associated with the requested signing profile, the value of alias will be "default" and for selecting using
user keys based on users email address, the alias should be "fromEmailAddress". This instructs the ADSS
Server to use the unique user key which has been registered for this user under that email address.
Password is optional attribute and only required if certificate was generated using the ADSS Certification
Service and an HSM is not being used (not in case of HSM is being used then the HSM is responsible for
the secure protection of the signing keys)..
-->
<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
<SignEmail alias="fromEmailAddress" />

```

```
<!-- this is the alias of the key to be used for signing the email -->

</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>
```

XML Tags	Description
OUTGOING	The outgoing element specifies that all the elements under this will be used to process outgoing emails.
EMAIL_SIGNED	Indicates whether email is signed (true) or not (false).
ENCRYPTED	Indicates email is encrypted (true) or not (false).
ATTACHMENT_CONTAINS	Indicates attachment is of type pdf, xml, doc etc.
SIGNED	Indicates whether the attachment is signed (true) or not (false).
OPERATION_RULE	The OperationRule element defines sequence of operations that the Secure Email Server Maillet will perform in one workflow. The OperationRule element contains one of the following values <ul style="list-style-type: none"> • SIGN_ATTACHMENT • SIGN_EMAIL • SIGN_ATTACHMENT,SIGN_EMAIL • SIGN_EMAIL,ARCHIVE_EMAIL • SIGN_EMAIL,ENCRYPT_EMAIL • SIGN_ATTACHMENT,SIGN_EMAIL,ARCHIVE_EMAIL • SIGN_ATTACHMENT,SIGN_EMAIL,ENCRYPT_EMAIL • SIGN_ATTACHMENT,ARCHIVE_EMAIL • SIGN_ATTACHMENT,ENCRYPT_EMAIL
SIGN_ATTACHMENT_PROFILE_ID	It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment.
OVERRIDABLE_ATTRIBUTES	PDF signature appearance elements which are being overridden from the default values defined in signing profile. Following are overridable attributes: Signing Reason Signing Location Signing Area Signing Page Signing Field Visibility Contact Information
SIGN_EMAIL_PROFILE_ID	It is the ID of the signing profile configured in ADSS Server Signing Service to sign an email.
SIGN_ATTACHMENT_ALIAS	It is the alias of the certificate configured in ADSS Server Key Manager module to sign an attachment.
SIGN_EMAIL_ALIAS	It is the alias of the certificate configured in ADSS Server Key Manager module to sign an attachment. It can also be an email address configured in ADSS Server Key Manager Service.
And	The emails will be filtered based on AND of all the fields under

XML Tags	Description
	this Element i.e. an email would be filtered if all conditions are met. Secure Email Server does not support mix of and/or filtering.
Or	The emails will be filtered based on OR of all the fields under this Element i.e. an email would be filtered if one of the criterions is met. Secure Email Server does not support mix of and/or filtering.

```

<Incoming>
<Verify>
<Workflow>
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="user1@companyABC.com" />
<From contains="user2@companyABC.com" />
<Subject contains="verify" />
<Body contains="confidential" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" />
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule attachResponseXml="true">ADD_RESULT_TO_TOP</VerificationResultRule>
<!-- The VerificationResultRule defines how to handle the verification results. -->
</Workflow>
</Verify>

<Verify>
<Workflow>
<Filter>
<!--The emails will be filtered based on the fields under this element.-->
<Or>
<To contains="user1@companyABC.com" />
<From contains="user2@companyABC.com" />
<CC contains="user3@companyABC.com" />
<Subject contains="verify" />
<Body contains="confidential" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" />
</Or>

</Filter>

<OperationRule>VERIFY_EMAIL</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_ORIGINAL_EMAIL_AS_ATTACHEMENT</VerificationResultRule>
</Workflow>
</Verify>

```

```
</Incoming>
</SES>
```

XML Tags	Description
INCOMING	The incoming element specifies that all the elements under this will be used to process incoming emails.
ATTACH_RESPONSE_XML	It has possible values of true and false. It appends the results of signature verification as an XML file if value is set to true.
OPERATION_RULE	The OperationRule element defines sequence of operations that a maillet perform in one workflow. The OperationRule element contains one of the following values: <ul style="list-style-type: none"> • VERIFY_ATTACHMENT • VERIFY_EMAIL • VERIFY_ATTACHMENT,VERIFY_EMAIL
VERIFICATION_RESULT_RULE	The VerificationResultRule defines how to handle the verification results. Possible options are: <ul style="list-style-type: none"> • ADD_RESULT_TO_TOP (Verify Attachment) • ADD_RESULT_TO_BOTTOM (Verify Attachment) • ADD_ORIGINAL_EMAIL_AS_ATTACHMENT (Verify Email)
And	The emails will be filtered based on AND of all the fields under this Element i.e. an email would be filtered if all conditions are met. Secure Email Server does not support mix of and/or filtering.
Or	The emails will be filtered based on OR of all the fields under this Element i.e. an email would be filtered if one of the criterions is met. Secure Email Server does not support mix of and/or filtering.



It is very important to note that restarting of the Ascertia-SES service is essential whenever any change is made in SES.xml configuration file.



Note the filter criteria for two workflows must not be same. If a filter criterion is met the Secure Email Server does not check any further filters and start processing the email based on the first matched workflow.

7 Example Configurations

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

7.1 Example-1

This is an example of signing PDFs attachments only where an existing blank signature field needs to be signed through ADSS Signing service profile, with no overrideable attributes. Assuming email could be from only accounts@companyABC.com being sent to anyone and there is a specific signing key for this department.

```
<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<Filter>
<And>
<From contains="accounts@companyABC.com" />
<To contains="*@*.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
<SignAttachment profileId="adss:signing:profile:001">
<!--
```

It is the ID of the signing profile configured in ADSS Server Signing Service to sign a PDF in an existing blank signature field present on the PDF document. The name of the blank signature filed is specified in the

```

signing profile.
-- >
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>
</SES>

```

7.2 Example-2

This is an example of signing XML attachments. Assuming email could be from only accounts@companyABC.com being sent to anyone and there is a specific signing key for this department. At the receiving end, attachment of the email will be verified and results will be added at the bottom of the email body. Also, verification response will be added as an xml attachment at receiving end.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<Filter>
<And>
<From contains="accounts@companyABC.com" />
<To contains="*@*.com" />

```

```

<Email signed="false" encrypted="false" />
<Attachment contains="xml" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
<SignAttachment profileId="adss:signing:profile:003">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an XML document.
-->
</SignAttachment>
</ProfileSelection>
<CertificateSelection>
<SignAttachment alias="bob" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="*@*.com" />
<From contains="accounts@companyABC.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="xml" signed="true" />
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_RESULT_TO_BOTTOM</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.3 Example-3

This is an example of signing pdf attachments only where signing is done at specific location (i.e. Center of the page) with signing location and signing reason as overrideable attribute. Assuming email being sent from accounts@companyABC.com in one work flow and from accounts2@companyABC.com in other work flow. In both cases, email is sent to anyone. Signing of pdf attachment is done using each user's unique key referenced from the emails From field. At the receiving end, attachment of the email will be verified and results will be added at the bottom of the email body. Also, verification response will be added as an xml attachment at receiving end.

```
<SES>
```

```

<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<!-- work flow #1 which will filter emails sent from accounts@companyABC.com having pdf file as an
attachment -->
<Filter>
<And>
<From contains="accounts@companyABC.com" />
<To contains="*@*.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
<SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment. This
profile specifies a signing area. Also signing reason and signing location are overrideable attributes. -->
<OverrideableAttributes>
<Attribute>
<Name>SIGNING_LOCATION</Name>
<Value>Surrey, UK</Value>
</Attribute>

<Attribute>
<Name>SIGNING_REASON</Name>
<Value>I certify this document</Value>

```

```

</Attribute>
</OverrideableAttributes>
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="fromEmailAddress" />
<!-- the alias of the key to be used for signing the attachment will be used based on from Email Address
field -->
  </CertificateSelection>
</Workflow>
</Sign>

<Sign>
<Workflow>
<!-- work flow #2 which will filter emails sent from accounts2@companyABC.com having pdf file as an
attachment -->
<Filter>
<And>
  <From contains="accounts2@companyABC.com" />
  <To contains="*@*.com" />
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:003">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment. This
profile specifies a signing area. Also signing location and signing reason are overrideable -- >
<OverrideableAttributes>
<Attribute>
<Name>SIGNING_LOCATION</Name>
<Value>Surrey, UK</Value>
</Attribute>

<Attribute>
<Name>SIGNING_REASON</Name>
<Value>I verify this document</Value>
</Attribute>
</OverrideableAttributes>
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="fromEmailAddress" />
<!-- the alias of the key to be used for signing the attachment will be used based on from Email Address field
-->
  </CertificateSelection>

```

```

</Workflow>
  </Sign>
  </Outgoing>

<Incoming>
<Verify>
<Workflow>
<!-- work flow #1 which will verify emails received from accounts@companyABC.com having pdf file as an
attachment to be verified-->
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="*@*.com" />
<From contains="accounts@companyABC.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_RESULT_TO_BOTTOM</VerificationResultRule>
</Workflow>
</Verify>

<Verify>
<Workflow>
<!-- work flow #2 which will verify emails received from accounts2@companyABC.com having pdf file as an
attachment to be verified-->
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="*@*.com" />
<From contains="accounts2@companyABC.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_RESULT_TO_BOTTOM</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.4 Example-4

This is an example of signing pdf attachments only where signing is done on bottom left position at page 2 of the document. All attributes such as signing location, signing reason, signing page, signing area, contact information and visibility is marked as overrideable attributes. Assuming email being sent from accounts@companyABC.com to quest@companyABC.com. At the receiving end, attachment of the email will

be verified and results will be added at the bottom of the email body. Also, verification response will be added as an xml attachment at receiving end.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>

<Workflow>
<Filter>
<And>
<From contains="accounts@companyABC.com" />
<To contains="guest@companyABC.com" />
<CC contains="*@*.com" />
<BCC contains="*@*.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
<SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign pdf attachment. All
signing attributes e.g. signing reason, signing location, contact information, signing page and visibility are
marked as overridable
-- >
<OverrideableAttributes>
<Attribute>
<Name>SIGNING_LOCATION</Name>

```

```
<Value>Surrey, UK</Value>
</Attribute>

<Attribute>
<Name>SIGNING_REASON</Name>
<Value>I certify this document</Value>
</Attribute>

<Attribute>
<Name>SIGNING_PAGE</Name>
<Value>2</Value>
<!--Defines on which page number to sign in the document -->
</Attribute>

<Attribute>
<Name>SIGNING_AREA</Name>
<Value>4</Value>
<!--Possible values are: 1 for Signing at TopLeft location, 2 for Signing at TopRight location, 3 for Signing at
Center location, 4 for Signing at BottomLeft location, 5 for Signing at Bottom Right location-->
</Attribute>

<Attribute>
<Name>CONTACT_INFO</Name>
<Value>111-222-333</Value>
</Attribute>

<Attribute>
<Name>VISIBILITY</Name>
<Value>true</Value>
</Attribute>

</OverrideableAttributes>
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="fromEmailAddress" />
<!-- the alias of the key to be used for signing the attachment will be used based on from Email Address field
-->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="guest@companyABC.com" />
```

```

<From contains="accounts@companyABC.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_RESULT_TO_BOTTOM</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.5 Example-5

This is an example of signing pdf attachments only where signing is done on bottom right at page 2 of the document. All attributes such as signing location, signing reason, signing page, signing area, contact information and visibility (signature visibility is marked as invisible) is marked as overrideable attribute. Assuming email being sent from accounts@companyABC.com to guest@companyXYZ.com. At the receiving end, attachment of the email will be verified and results will be added at the top of the email body. Also, verification response will be added as an xml attachment at receiving end.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<Filter>
<And>
<From contains="accounts@companyABC.com" />

```

```

<To contains="guest@companyXYZ.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
<SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign a pdf attachment. All
signing attributes e.g. signing reason, signing location, contact information, signing page and visibility are
marked as overridable
-->
<OverrideableAttributes>
<Attribute>
<Name>SIGNING_LOCATION</Name>
<Value>Surrey, UK</Value>
</Attribute>

<Attribute>
<Name>SIGNING_REASON</Name>
<Value>I certify this document</Value>
</Attribute>

<Attribute>
<Name>SIGNING_PAGE</Name>
<Value>2</Value>
<!--Defines on which page number to sign in the document -->
</Attribute>

<Attribute>
<Name>SIGNING_AREA</Name>
<Value>5</Value>
<!--Possible values are: 1 for Signing at TopLeft location, 2 for Signing at TopRight location, 3 for Signing at
Center location, 4 for Signing at BottomLeft location, 5 for Signing at Bottom Right location-->
</Attribute>

<Attribute>
<Name>CONTACT_INFO</Name>
<Value>111-222-333</Value>
</Attribute>
<Attribute>
<Name>VISIBILITY</Name>
<Value>>false</Value>
</Attribute>

</OverrideableAttributes>
</SignAttachment>
</ProfileSelection>

<CertificateSelection>

```

```

<SignAttachment alias="fromEmailAddress" />
<!-- the alias of the key to be used for signing the attachment will be used based on from Email Address field -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<Filter>
<!--The emails will be filtered based on all the fields under this element.-->
<And>
<To contains="guest@companyXYZ.com" />
<From contains="accounts@companyABC.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule
attachResponseXml="true">ADD_RESULT_TO_BOTTOM</VerificationResultRule>

</Workflow>
</Verify>
</Incoming>
</SES>

```

7.6 Example-6

This is an example of signing PDFs attachments only where an existing blank signature field needs to be signed, with no overrideable attributes. Assuming email being sent from UK department accountsUK@companyABC.com and a FR department accountsFR@companyABC.com in different work flows to anyone and each department has their own signing profile on ADSS Server for signing the attachment. At the receiving end, attachment of the email will be verified and results will be added at the top of the email body.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>

```

```

<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<!-- work flow #1 which will send emails from accountsUK@companyABC.com having pdf file as an
attachment -->
<Filter>
<And>
  <From contains="accountsUK@companyABC.com" />
  <To contains="*@*.com" />
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment. None of
the signing attributes is overrideable.
-->
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>

<Sign>
<Workflow>
<!-- work flow #2 which will send emails from accountsFR@companyABC.com having pdf file as an
attachment -->
<Filter>
<And>
  <From contains="accountsFR@companyABC.com" />
  <To contains="*@*.com" />
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>

```

```

<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:002">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign a PDF attachment.
None of the signing attributes is marked overrideable
-- >
  </SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="bob" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<!-- work flow having an OR tag so that It can verify the attachment of emails coming from any domain/user
-- >
<Filter>
<Or>
<To contains="*@*.com" />
<From contains="*@*.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</Or>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule attachResponseXml="false">ADD_RESULT_TO_TOP</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.7 Example-7

This is an example of signing PDFs attachments only where an existing blank signature field needs to be signed, with no overrideable attributes. Assuming email being sent from UK department accountsUK@companyABC.com (with having sign keyword in the subject body and confidential key word in the email body) and a FR department accountsFR@companyABC.com (with having certify keyword in the subject body and secure key word in the email body) being sent to anyone and each department has their own signing profile on ADSS Server for signing the attachment. At the receiving end, attachment of the email will be verified and results will be added at the top of the email body.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

```

```

<SigningServiceSettings>
  <SigningServiceAddress>
    <PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
    <SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
  </SigningServiceAddress>
  <NoFilterAction>EMAIL_TO_ADMINISTRATOR</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
  <VerificationServiceAddress>
    <PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
    <SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
  </VerificationServiceAddress>
  <NoFilterAction> EMAIL_TO_ADMINISTRATOR </NoFilterAction>
  <ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<!-- work flow #1 which will send emails from accountsUK@companyABC.com having pdf file as an
attachment. Also, having sign and confidential keywords as filter criteria -->
<Filter>
<And>
  <From contains="accountsUK@companyABC.com" />
  <To contains="*@*.com" />
  <Subject contains="sign"/>
  <Body contains="confidential"/>
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment.
-->
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>

<Sign>
<Workflow>

```

```

<! -- work flow #2 which will sent emails from accountsFR@companyABC.com having pdf file as an
attachment. Also, having certify and secure keywords as filter criteria -->
<Filter>
<And>
  <From contains="accountsFR@companyABC.com" />
  <To contains="*@*.com" />
  <Subject contains="certify"/>
  <Body contains="secure"/>
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:002">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign an attachment.
-->
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="bob" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<!--
work flow having an OR tag so that It can verify the attachment of emails coming from any domain/user -->
<Filter>
<Or>
  <To contains="*@*.com" />
  <From contains="*@*.com" />
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="true"/>
</Or>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule attachResponseXml="false">ADD_RESULT_TO_TOP</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.8 Example-8

This is an example of signing PDFs attachments only where an existing blank signature field needs to be signed. The existing blank signature field is the only attribute marked as override able. Assuming email being sent from any one and addressed to any one. At the receiving end, attachment of the email will be verified and results will be added at the top of the email body.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
<NoFilterAction>EMAIL_TO_RECIPIENTS</NoFilterAction>
</SigningServiceSettings>
<VerificationServiceSettings>
<VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
<NoFilterAction> EMAIL_TO_ADMINISTRATOR_AND_TO_RECIPIENT </NoFilterAction>
<ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
<EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<Filter>
<And>

  <From contains="*@*.com" />
  <To contains="*@*.com" />
  <Subject contains="sign"/>
  <Body contains="confidential"/>
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign a PDF attachment. This
profile specifies a blank signature field to be signed and this is also overrideable. -->
<OverrideableAttributes>

```

```

<Attribute>
<Name>SIGNING_FIELD</Name>
<Value>Signature20</Value>
<!-- Defines the name of existing blank signature field already present in the pdf to be signed in ADSS
Signing profile -->
</Attribute>
</OverrideableAttributes>
</SignAttachment>
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<Filter>
<And>
<To contains="*@*.com" />
<From contains="*@*.com" />
<Email signed="false" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT</OperationRule>
<VerificationResultRule attachResponseXml="false">ADD_RESULT_TO_TOP</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>

```

7.9 Example-9

This is an example of signing PDFs attachments only where an existing blank signature field needs to be signed. The existing blank signature field is the only attribute marked as override able. Assuming signed email is being sent from any one and addressed to any one. At the receiving end, attachment of the email and email it self will be verified and results will be added at the top of the email body.

```

<SES>
<Settings>
<DomainName>companyABC.com</DomainName>
<AdminEmailAddress>admin@companyABC.com</AdminEmailAddress>
<OriginatorId>ADSS_QAS</OriginatorId>

<SigningServiceSettings>
<SigningServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/signing/dsi</PrimaryAddress>

```

```

<SecondaryAddress>http://machine2:8777/adss/signing/dsi</SecondaryAddress>
</SigningServiceAddress>
  <NoFilterAction>EMAIL_TO_RECIPIENTS</NoFilterAction>
</SigningServiceSettings>
  <VerificationServiceSettings>
  <VerificationServiceAddress>
<PrimaryAddress>http://machine1:8777/adss/verification/svi</PrimaryAddress>
<SecondaryAddress>http://machine2:8777/adss/verification/svi</SecondaryAddress>
</VerificationServiceAddress>
  <NoFilterAction> EMAIL_TO_ADMINISTRATOR_AND_TO_RECIPIENT </NoFilterAction>
  <ResponseNotTrustedAction> EMAIL_TO_ADMINISTRATOR </ResponseNotTrustedAction>
</VerificationServiceSettings>
  <EmailErrorAction> EMAIL_TO_ADMINISTRATOR </EmailErrorAction>
</Settings>

<Outgoing>
<Sign>
<Workflow>
<Filter>
<And>

  <From contains="*@*.com" />
  <To contains="*@*.com" />
  <Subject contains="sign"/>
  <Body contains="confidential"/>
  <Email signed="false" encrypted="false" />
  <Attachment contains="pdf" signed="false" />
</And>
</Filter>
<OperationRule>SIGN_ATTACHMENT, SIGN_EMAIL</OperationRule>
<ProfileSelection>
  <SignAttachment profileId="adss:signing:profile:001">
<!--
It is the ID of the signing profile configured in ADSS Server Signing Service to sign a PDF attachment. This
profile specifies a blank signature field to be signed and this is also overrideable. -->
<OverrideableAttributes>
<Attribute>
<Name>SIGNING_FIELD</Name>
<Value>Signature20</Value>
<!-- Defines the name of existing blank signature field already present in the pdf to be signed in ADSS
Signing profile -- >
</Attribute>
</OverrideableAttributes>
</SignAttachment>
  <SignEmail profileId="adss:signing:profile:002">
</ProfileSelection>

<CertificateSelection>
<SignAttachment alias="alice" />
<!-- this is the alias of the key to be used for signing the attachment. -->
<SignEmail alias="alice" />

```

```
<!-- this is the alias of the key to be used for signing the email. -->

</CertificateSelection>
</Workflow>
</Sign>
</Outgoing>

<Incoming>
<Verify>
<Workflow>
<Filter>
<And>
<To contains="*@*.com" />
<From contains="*@*.com" />
<Email signed="true" encrypted="false" />
<Attachment contains="pdf" signed="true"/>
</And>
</Filter>
<OperationRule>VERIFY_ATTACHMENT, VERIFY_EMAIL</OperationRule>
<VerificationResultRule attachResponseXml="false">ADD_RESULT_TO_TOP</VerificationResultRule>
</Workflow>
</Verify>
</Incoming>
</SES>
```