ADSS Verification Service

Table of Contents

How can I setup ADSS Verification Service to verify signatures produced using a specific PKI?

The following checklist should be considered when setting up ADSS Verification Service to verify signatures produced using a specific PKI:

1. The relevant CA certificates should be registered in **Trust Manager** module
2. Verification Service Profile **Trust Anchor** page should be updated to include newly registered CAs
3. Verification Profile should be assigned to relevant client in the **Client Manager** module in case of new profile
4. CRLs should be downloaded for the CAs within CRL Monitor > **CRL Details** page
5. ADSS Server should be restarted from **Server Manager** module after making above changes

How to configure ADSS Verification Service to verify digital signatures?

The following checklist needs to be configured to verify digital signatures:

1. Register the relevant CA's certificates in Trust Manager, see details.
2. Create a new Verification Service Profile and configure the newly registered CA in it, see details. In case the verification profile already exists, then edit it and click the "Trust Anchors Settings" tab to include the newly registered CA, see details.
3. In case Verification Profile is not assigned, assign it to the relevant client in the Client Manager module by clicking the "Verification Service" tab, see details.
4. Restart ADSS Server from Server Manager module after making above changes.
5. Start sending the signature verification requests to ADSS Verification Service.

How to decide between the Basic or Advanced certificate path validation (RFC 5280 compliant) settings in a verification profile?

It depends on the business requirements of the client, whether they are fine with basic path validation or looking for a RFC 5280 compliant validation.

Basic Validation is used when policy processing and name constraints are not required to be checked in the certificate's validation. The path validation is performed by using Ascertia's custom algorithm, which is much faster and performs the following checks:

- Name Chaining
- Signature Verification
- Basic Constraints Verification
- Key Usages and Extended Key Usages Verification
- Revocation Status Check using CRL or OCSP

Advanced Validation is fully RFC 5280 compliant, and is mostly used when certificates (to validate) are issued by Federal PKI.

How to ensure seamless processing of signature/ certificate validation request, when the CA chain is already registered in Trust Manager?

It is often observed that CAs are registered in Trust Manager but they are not made available in the verification profile. Consequently, the validation request gets failed as the certificate path could not be properly built.

To resolve this:

1. Edit the Verification profile.
2. Go to the "Trust Anchors Settings" tab.
3. Place the CAs in the Allowed CAs List that will be used to build the path.
4. Save the settings.
5. Restart the Verification service from the Service Manager module and resend the request.

How to invoke Verification profile for non-registered CAs?

When the target certificate chain is built up to the registered self-signed Root CA certificate, and the intermediate CA certificate(s) are not registered within the Trust Manager module, then the revocation of such non-registered CAs is discovered by using a non registered CA policy (configured within the Verification Profile > Advanced Settings). ADSS Server also provides the flexibility to choose a certificate validation mechanism from CDP, AIA and configured OCSP addresses.

How to enhance existing signatures by using ADSS Verification Service?

The Verification Service can also be used to enhance the existing signatures to more advanced signatures, as part of a validation process. The Verification Service has implemented "Advanced Electronic Signature (AdES)" profile of the OASIS DSS, and can enhance the basic CAdES, XAdES and PAdES signatures to their relevant advanced formats i.e. -X, -T, -C, -XL and -A signatures. For this:

1. Go to ADSS-Client-SDK/ API/ apidocs/ adss/ index.html
2. For more details, see the "setReturnUpdatedSignature()" method of the "com.ascertia.adss.client.api.verification.SignatureVerificationRequest" class.