ReleaseNotes

Version 0.10.0 (next feature release)

Release date: TBA

For a complete list of issues addressed in this release, see https://issues.shibboleth.net/jira/issues/?filter=12070

This is a major pre-1.0 feature release.

Dependency Changes

- MDA-217: This release is built with Java 11, and requires a Java 11 or later execution environment. (Previously MDA-189 targeted the release to J ava 8)
- JPAR-125: Updated from V4.3 to V5.1 of Spring Framework.

Packaging Changes

- MDA-158: The packaging of the RSA key blacklist resources introduced in version 0.9.0 has been changed. Previously included in the `aggregat or-pipeline` artifact, these resources have now been moved into a separate `aggregator-blacklists` artifact. The resource names have not changed. This means that if your application does not use these resources, it may decrease in size by around 13MB. Applications making use of the blacklist resources may need to add a dependency on `aggregator-blacklists`.
- MDA-181: A Maven BOM (Bill Of Materials) artifact has been made available. This makes it easier for projects using the Shibboleth MDA as a dependency to acquire a consistent set of managed dependencies without using the Shibboleth parent POM. You can include the MDA BOM in your Maven project like this:

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>net.shibboleth.metadata</groupId>
            <artifactId>aggregator-bom</artifactId>
            <version>0.10.0</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```

- JPAR-113: The packaged `aggregator-`*`module`* JAR files, as well as a number of underlying library JARs have been given automatic module names as described in Java Modularity, for future use on the Java module path under Java 9 and later releases. Note that this version of the Shibboleth Metadata Aggregator has not been tested on, and is therefore not guaranteed to work on, the Java module path.

API Additions

- MDA-52: A new stage `EntitiesStrippingStage` has been added to allow stripping a number of different elements (all from the same namespace) from a DOM document. The stage may be operated in a blacklisting or whitelisting mode, with blacklisting the default. Like `EntitySt rippingStage`, an `elementNamespace` property determines the namespace in question, and all elements in other namespaces are ignored.
- MDA-56: A new stage `EntityAttributeAddingStage` has been added to add entity attributes to the metadata for SAML entities. This is configured using `attributeName`, `attributeNameFormat` and `attributeValue` properties, with `attributeName` and `attributeNameFo rmat` defaulting to the values required to add an entity category attribute. The stage is based on a new `Container` framework which attempts to generate reasonably well formatted XML for nested container elements, and handles the insertion of the required parent containers (`Extensions`, `EntityAttributes`, `Attribute`) when they are not already present.
- MDA-160: The `EntityAttributesFilteringStage` has been extended with a new `recordingRemovals` property, defaulting to false. If `rec ordingRemovals` is set to `true`, each removed entity attribute is recorded as a `WarningStatus` in the item's item metadata, indicating the name and value of the entity attribute removed. This can then be processed by subsequent stages, such as a `StatusMetadataLoggingStage`.
- MDA-177: An entity attribute matcher `AssuranceCertificationMatcher` has been added to allow simpler matching of entity attributes containing assurance certifications, such as that used by the SIRTFI framework.
- MDA-178: A bean definition resource has been added to simplify access to each bean class in the aggregator-pipeline artifact. In XML configuration, this can be accessed by `<import resource="classpath:net/shibboleth/metadata/beans.xml"/>`. One abstract bean is defined for each available bean class, named after the class's simple name prefixed by "`mda.`". After including this resource, for example, `clas s="net.shibboleth.metadata.dom.XMLSignatureValidationStage"` can be replaced by `parent="mda. XMLSignatureValidationStage"`; this definition will also include the `init-method` and `destroy-method` properties for the bean when appropriate.
- MDA-179: The simple command-line interface now includes a `--version` option to request the printing of the framework version number.
- MDA-184: A new utility class `RegexFileFilter` has been added to support one of the common use cases of the `DOMFilesystemSourceStage`, where only certain files should be processed from a directory, based on their names.
- MDA-193: To make using the `Validator` framework more straightforward, the new `ValidatorSequence` class abstracts the concept of a sequence of `Validators` which can be maintained and applied as a group. Existing classes requiring this behaviour have been refactored to take advantage of `ValidatorSequence`.

- MDA-199: A new `X509ROCAValidator` component allows RSA public keys in X.509 certificates to be checked for vulnerability to ROCA (the Return of Coppersmith's Attack, also known as CVE-2017-15361).
- MDA-200: The `BaseValidator` abstract class has been extended to add an `addErrorMessage` method and a `message` property, which acts as a format string for `ErrorStatus` item metadata generated through `addErrorMessage`.
- MDA-201: New `AcceptAllValidator` and `RejectAllValidator` components have been added. Both always return `Action.DONE` so that they can be used to terminate a sequence of validators. `AcceptAllValidator` has no other functionality; `RejectAllValidator` uses its `message` property to format an appropriate `ErrorStatus` for the `Item` on which validation is being performed.
- MDA-202: Four new validator components (`AcceptStringValueValidator`, `RejectStringValueValidator`, `AcceptStringRegexValidator` and `RejectStringRegexValidator`) have been added to match `String` values. All four return `Action.DONE` if the match occurs and will therefore terminate a sequence of validators; `Action.CONTINUE` is returned otherwise. The `Reject` forms also add a formatted `ErrorStatus` on matching.
- JSPT-73: This release bundles a new version of the Shibboleth `java-support` package, which implements a new `FixedStringIdentifierGenerationStrategy` for use when it is not necessary to use different `ID` attribute values for different documents.
- MDA-214: A new `X509DSADetector` component allows DSA keys in metadata to be rejected, or merely warned about.

API Changes

- MDA-166: The `ItemSerializer` and `ItemCollectionSerializer` interfaces now allow serializers to throw `IOException` when appropriate. The provided `DOMItemSerializer` will throw an `IOException` wrapping a `TransformerException` if the latter is thrown during XML serialization. Previously, this condition would only have resulted in logging at ERROR level.
- MDA-167: The `ItemIdTransformStage` now transforms identifiers using a collection of `Function` objects rather than of the similar `Converter` provided by the Spring framework. This also affects the type of the `MDQueryMD5ItemIdTransformer` and `MDQuerySHA1ItemIdTransformer` classes. This change will not affect existing configurations if only those classes are in use. This matches the use of `Function` elsewhere in the API, and allows the use of Guava's `Functions` helper class.
- MDA-169: The `SAMLMetadataSupport.getDescriptorExtensions` method has been renamed to `getDescriptorExtension` to reflect the fact that it returns a single result.
- MDA-171: The `SAMLMetadataSupport.getDescriptorExtension` method's parameters must now be non-`null`; their annotations have been changed to `@Nonnull` to correspond with this. In previous releases, they were annotated as `@Nullable` and passing `null` would result in the method returning `null`.
- MDA-175: The `ItemOrderingStrategy` interface defined by the `EntitiesDescriptorAssemblerStage` now allows the ordering strategy to throw a `StageProcessingException` if, for example, the items presented are invalid in some way and can not be ordered. Such an exception will be propagated upwards to the caller of the stage's `execute` method.
- MDA-179: The `Version` class's `getMicroVersion` method has been renamed to `getPatchVersion` to align with current (semantic versioning) terminology.
- MDA-182: Several classes exposed as part of the API for building custom stages have been reworked to simplify implementation of other stages and to correspond to current naming conventions:
  - `BaseStage` has been renamed to `AbstractStage`
  - `BaseIteratingStage` has been renamed to `AbstractFilteringStage`
  - A new `AbstractIteratingStage` allows the simpler construction of stages which process each Item independently
- MDA-188: The `AbstractDOMTraversalStage` framework has been generalised to allow the use of custom context objects specific to the particular traversal, rather than relying on sometimes tortured uses of the `ClassToInstanceMultiMap` to carry everything. This is a breaking change, but will only affect writers of stages derived from `AbstractDOMTraversalStage`:
  - Context objects must implement the `DOMTraversalContext` interface. This no longer includes the `getStash` method (returning a `ClassToInstanceMultiMap` but does add a new `end()` method to be called at the end of the traversal.
  - A basic implementation of `SimpleDOMTraversalContext` is provided without any data fields. This can be used in many cases where custom storage is not required in the context; for an example, see `AbstractElementVisitingStage`.
  - More complex cases can extend `SimpleDOMTraversalContext` to include additional fields and method. For a very straightforward example, see `CRDetectionStage`. A more complex example, including use of the `end()` method from `DOMTraversalContext`, can be found in `ElementsStrippingStage`.
- MDA-192: The `ancestorEntity` method has been removed from `AbstractDOMTraversalStage`; a protected `errorPrefix` method has replaced it in order to allow sub-classes to replicate this or similar behaviour. A new `AbstractSAMLTraversalStage` class has been added to incorporate the specific old behaviour.
- MDA-198: In previous releases, the three X.509 validation component (`X509RSAExponentValidator`, `X509RSAKeyLengthValidator` and `X509RSAOpenSSLBlacklistValidator`) all set a default ID related to their names (e.g., `RSAKeyLength`). This default ID setting behaviour has been removed. This may have two effects on configurations which do not explicitly set the component ID:
  - If a configuration did not set the component ID, initializing the component will now fail with a `ComponentInitializationException`.
  - Configurations that implicitly set the component ID to a defaulted Spring component ID using `IdentifiableBeanPostProcessor` may give different results, as the Spring component ID may now appear in status objects replacing the previous default.
- MDA-206: The `PipelineDemultiplexerStage`'s `waitingForPipelines` property previously defaulted to `false`, which could result in unexpected behaviour if the stage was invoked a second time without arranging to synchronise execution with the called pipelines. As a result, most deployments set `waitingForPipelines` to `true` so that the called pipelines will all complete before control is passed on from the `PipelineDemultiplexerStage`; this behaviour is now the default.
- JSE-28: This release bundles a new version of the Shibboleth `spring-extensions` project, which removes support for SVN-based resources.
- MDA-191: The stages `PullUpCacheDurationStage`, `PullUpValidUntilStage`, `SetCacheDurationStage`, `SetValidUntilStage` and `ValidateValidUntilStage` now use the `Instant` and `Duration` classes (introduced in Java 8) in their APIs rather than using `long` values representing milliseconds as in previous releases.
  - This aligns the metadata aggregator with other Shibboleth projects based on the Java 11 platform. If you use the Java language to configure these stages, you will need to re-code appropriately; in most cases, this can be done quickly using `Instant.fromEpochMilli()` and `Duration.ofMillis()`, but we recommend adopting the modern `java.time` classes throughout.
  - The `DurationToLongConverter` is no longer included as part of the `java-support` dependency. If you were using it as part of an XML configuration to specify durations in ISO 8601 format (e.g., "PT15M") then you should replace references to `DurationToLongConverter` with references to the new `StringToDurationConverter`.
- MDA-222: The contract for bean properties representing collections has changed:
  - Property setters for collection properties are now annotated as `@Nonnull` `@NonnullElements` `@Unmodifiable`.
    - Previously, some setters allowed a null value to act in place of an empty collection. This usage will now result in most cases in a `NullPointerException`.

        - Previously, some setters filtered null values out of provided collections. Again, this usage will now result in most cases in a `NullPointerException`.
        - Setters now guarantee not to modify the passed collection. This was previously true in practice in most cases, but is now guaranteed.
    - Most property getters for collection properties are also now annotated as `@Nonnull` `@NonnullElements` `@Unmodifiable`.
        - In exceptional cases, getters may be annotated as `@NonnullAfterInit` instead of `@Nonnull`. This is only done when an "empty collection" default is inappropriate for the property and would normally be accompanied by `@NotEmpty` on both the setter and the getter.
- MDA-223: A number of constant fields have been removed from the `XMLSignatureSigningStage` and `XMLDSIGSupport` classes and therefore the API, as they are now part of the base Java API. For example, `XMLSignatureSigningStage.ALGO_ID_SIGNATURE_RSA_SHA256` is replaced by Java's `SignatureMethod.RSA_SHA256`.

Improvements

- MDA-183: the `compromised-1024.txt` and `compromised-2048.txt` resources have been extended with keys shipped with some releases of the Jetty container.

Bug Fixes

- MDA-179: The `Version` class is now functional, rather than throwing a `NullPointerException` when used.
- MDA-196: Setting the `XMLSignatureSigningStage`'s `includeX509SubjectName` property to `true` caused a `ClassCastException`. It now behaves as intended, resulting in an `<X509SubjectName>` element being added to the signature's `<KeyInfo>`'s `<X509Data>` element.
- MDA-216: signatures generated by the `XMLSignatureSigningStage` under Java 11 are now consistent with signatures generated under earlier versions of Java.
- MDA-220: `EntityFilterStage` handled the case of whitelisting incorrectly when the collection of entity IDs to whitelist was empty. The stage now correctly removes all items from the collection, rather than removing none of them.
- MDA-224: `XMLSignatureSigningStage` threw an IndexOutOfBoundsException if the `includeKeyValue` property was set to `true` without either setting the `publicKey` or `certificates` properties; the stage now just omits the `KeyValue` from the signature as if `includeKeyValue` had been set to `false`.

Version 0.9.2 (current stable release)

Release date: 19th October 2016

This release adds some minor new features:

- MDA-76 multi-output serialiser for offline use cases
  This adds a `MultiOutputSerializationStage` which can be provided with a `Serializer` and an `OutputStrategy` to allow each `Item` in a collection to be serialized to a different location. This is intended for use cases such as per-entity metadata generation. A `FilesInDirectoryMultiOutputStrategy` is provided for this use case; its properties include a destination directory within which individual files are created based on a prefix and suffix string, and a transformed version of each item's first `ItemId`. Transformer classes `SHA1StringTransformer` and `PathSegmentStringTransformer` have been added to cover the most common current use cases. An example of the use of these new classes are available in this example.
- MDA-170 allow use of PKCS#11 for XML DSIG
  Adds a `PKCS11PrivateKeyFactoryBean` to allow a PKCS#11 token (such as a smart card or HSM) to be used to sign documents. An example of its use can be found in this example. Note that this class is deprecated and will not appear in version 0.10.0. In that release, the same functionality will be available from the spring-extensions project, see JSE-20.

The following bug fix is included:

- MDA-168 `EntityAttributeFilteringStage` mishandles multiple containers
  The `EntityAttributeFilteringStage` only processed the first `EntityAttributes` container in an entity descriptor's `Extensions`. Although the specification requires that at most one such container be present, this is not a schema constraint and cannot be relied on in security-sensitive applications. `EntityAttributeFilteringStage` now processes all `EntityAttributes` containers in an entity.