MAEC v4.1 Release Notes

February 11, 2014

MAEC v4.1 consists of four schemas:

- Version 4.1 of the MAEC Bundle schema
- Version 2.1 of the MAEC Package schema
- Version 2.1 of the MAEC Container schema
- Version 1.1 of the MAEC Default Vocabularies schema, which defines default controlled vocabularies used within MAEC v4.1

This represents a minor update release of MAEC, and thus v4.1 is backwards compatible with previous MAEC 4.x versions, with one minor exception as noted below.

New Feature Highlights

- Added support for Cyber Observables eXpression (CybOX) v2.1.
- Added implementation of MAEC Capabilities, for capturing the high-level capabilities that a malware instance may possess; please see High-level Changes below for more information.
- Added ability to "label" a Malware Subject with common terms, e.g. "worm".
- Added ability to capture details of configuration parameters used by a malware instance.
- Added ability to capture details of the development environment used in the development of a malware instance.
- Many vocabulary updates, tweaks, and changes.

High-Level Changes

- The MAEC Bundle, Package, and Container schemas were updated to incorporate Cyber Observable eXpression (CybOX™) v2.1.
- MAEC now offers a standard way of capturing the set of high-level abilities that a malware instance possesses, which we term Capabilities, via the Capabilities field at the root level of the MAEC Bundle. For instance, to state that a malware instance is capable of persisting, one may simply specify a single MAEC Capability with a name of "persistence." Furthermore:

    - Strategic Objectives and Tactical Objectives have been developed to more granularly capture the details of each Capability. Simply put, a Capability can have one or more Strategic Objectives that it attempts to carry out, and accordingly a Strategic Objective can have one or more Tactical Objectives in the same manner. Continuing with the persistence example, the malware instance could have a Strategic Objective of “persist to continuously execute on system,” which in turn could have a Tactical Objective of “persist after system reboot.” We’ve developed individual vocabularies for all Strategic and Tactical Objectives corresponding to a particular Capability, which are found in the MAEC Default Vocabularies schema; also available is a hierarchical view of the Capabilities and Objectives that we’ve added for this release, available here[1].

    - While Capabilities are intended to convey what a malware instance is capable of doing, there exists a clear link between Capabilities and the concrete ways they are implemented. We have supported this in MAEC by allowing for the linking between a Capability and/or one of its Objectives to one or more MAEC Behaviors that represent a particular implementation of the Capability or Objective.
- Restrictions on ID syntax have been lifted in all IDs used in MAEC types, so that all MAEC IDs are now compatible with the implementations used in the Cyber Observable eXpression (CybOX) and Structured Threat Information eXpression (STIX) languages. Consequently, the additional schematron and XSL files formerly used primarily for ID syntax validation have been deprecated.
- Many annotations in the Bundle and Package schemas have been fixed and updated based on the recently released MAEC v4.0.1 Specification.

Bundle Schema Changes

- Added `CapabilityType, CapabilityListType, CapabilityObjectiveType, CapabilityPropertyType, CapabilityRelationshipType, CapabilityReferenceType, CapabilityObjectiveReferenceType,` and `CapabilityObjectiveRelationshipType` for capturing malware Capabilities, their Objectives, and associated properties. Added `Capabilities` field, of `CapabilityListType`, to the `BundleType` as an implementation of this feature.
- Added `ordinal_position` attribute to the `ProcessTreeNodeType` for characterizing the ordinal position of the process with respect to other processes spawned or injected by the malware.
- Removed `ObjectIDPattern` and `ActionIDPattern`, as the Object/Action IDs they were patterning are not strictly enforceable in terms of schema validity through

[1] In the diagram, High-level Capabilities are in light blue, Strategic Objectives are in orange, and Tactical Objectives are in peach.

the usage of these types, due to their nature of being imported from the CybOX Core schema.

- Updated value of fixed `schema_version` attribute on `BundleType` to 4.1.
- For relaxing restrictions on ID syntax, changed the type of the ID attribute in the `BundleType, BehaviorType, BehaviorCollectionType, ActionCollectionType, ActionImplementationType, ObjectCollectionType, CandidateIndicatorType, ProcessTreeNodeType`, and `CandidateIndicatorCollectionType` to `xs:QName`.
- Due to relaxing of restrictions on ID syntax, removed extraneous ID pattern types: `BundleIDPattern, BundleIDREFPattern, BehaviorIDPattern, BehaviorIDREFPattern, ActionIDREFPattern, ObjectIDPattern, ActionImplementationIDPattern, CandidateIndicatorIDPattern, ActionCollIDPattern, BehaviorCollIDPattern, ObjectCollIDPattern, CandidateIndicatorCollIDPattern, ProcessTreeNodeIDPattern`, and `ActionEquivalenceIDREFPattern`.

Package Schema Changes

- Changed “manual” to “in-depth” in `AnalysisTypeEnum` for consistency with the other value in the enumeration[2].
- Changed multiplicity of `Findings_Bundle_Reference` field in `AnalysisType` to unbounded, to allow for multiple Bundles to be referenced from a single Analysis.
- Added `MalwareExceptionType` and implementation via `Raised_Exception` field in `DynamicAnalysisMetadataType` for capturing errors that may be raised and caught during the instrumented execution of malware in a sandbox or other dynamic analysis environment.
- Added `MalwareDevelopmentEnvironmentType` and implementation via `DevelopmentEnvironmentType` in `MalwareSubjectType` for capturing the tools and debugging files associated with the development of a malware instance.
- Added `Label` field to `MalwareSubjectType` for capturing common labels, e.g., 'worm', associated with a malware instance.
- Added `MalwareConfigurationDetailsType, MalwareConfigurationObfuscationAlgorithmType, MalwareConfigurationObfuscationDetailsType, MalwareConfigurationParameterType`, and `MalwareConfigurationStorageDetailsType` for capturing malware configuration parameters and their associated properties. Also, added a

[2] Not backwards compatible with previous MAEC 4.x versions

Configuration_Details field to the MalwareSubjectType as an implementation of this new capability.

- Added Compatible_Platform element of cyboxCommon:PlatformSpecificationType with an unbounded multiplicity to the MalwareSubjectType for capturing the platform(s) affected by a malware instance.
- Added observation_name attribute to CommentType for capturing the names/types/IDs of observations that may be made in analyst comments.
- Updated value of fixed schema_version attribute on PackageType to 2.1.
- For relaxing restrictions on ID syntax, changed the type of the ID attribute in the AnalysisType, PackageType, MalwareSubjectType, ActionEquivalenceType, and ObjectEquivalenceType to xs:QName. Similarly, changed types on *IDREF attribute in MalwareSubjectReferenceType to xs:QName.
- Due to relaxing of restrictions on ID syntax, removed extraneous ID pattern types: PackageIDPattern, PackageIDREFPattern, MalwareSubjectIDPattern, MalwareSubjectIDREFPattern, AnalysisIDPattern, ActionEquivalenceIDPattern, and ObjectEquivalenceIDPattern.

Container Schema Changes

- Updated value of fixed schema_version attribute on ContainerType to 2.1.
- For relaxing restrictions on ID syntax, changed the type of the ID attribute in the ContainerType to xs:QName.
- Due to relaxing of restrictions on ID syntax, removed extraneous ID pattern types: ContainerIDPattern.

Default Vocabulary Schema Changes

- Added MalwareLabelVocab/Enum for defining a default set of malware labels.
- Added MalwareConfigurationParameterVocab/Enum for defining a default set of malware configuration parameter names.
- Added MalwareDevelopmentToolVocab/Enum for defining a default set of types of tools used in malware development.
- Added (large) initial set of Malware Capability/Objective vocabularies for use in characterizing Malware Capabilities, their Objectives, and any associated properties. One vocabulary/enumeration, the MalwareCapabilityVocab/Enum, was added for the high-level Capabilities themselves. Accordingly, every capability has two corresponding vocabularies/enumerations: one for all relevant Strategic Objectives and another for Tactical Objectives, e.g. the

`PersistenceStrategicObjectivesVocab/Enum` and the `PersistenceTacticalObjectivesVocab/Enum` for the Objectives of the Persistence Capability.

- Updated the `MalwareSubjectRelationshipVocab/Enum` to version 1.1 as a result of adding several new values for the capture of a more diverse set of relationships between Malware Subjects. Note that the old vocabulary and enumeration, versioned at 1.0, has been retained in the schema for the sake of backwards compatibility.
- Updated the following Action name vocabularies/enumerations to version 1.1 as a result of adding several new values: `DeviceDriverActionNameVocab/Enum`, `LibraryActionNameVocab/Enum`, `DirectoryActionNameVocab/Enum`, `DiskActionNameVocab/Enum`, `FileActionNameVocab/Enum`, `NetworkActionNameVocab/Enum`, `UserActionNameVocab/Enum`, and `ServiceActionNameVocab/Enum`. Note that the old vocabularies and enumerations, versioned at 1.0, have been retained in the schema for the sake of backwards compatibility.

Other Changes

- Added example, package_capability_example, to demonstrate Capability/Objective usage and linkage between Capabilities, Behaviors, and Actions.
- Added example, package_capability_example_snifula, to provide a more detailed view of Capabilities/Objectives and their usage in characterizing a complex malware instance.
- Added example, package_development_environment_example, to demonstrate usage of the new malware development environment characterization ability.
- Added example, package_configuration_parameters_example, to demonstrate usage of the new malware configuration parameter characterization ability.
- Updated existing examples to better reflect MAEC best practices and add features introduced in MAEC 4.1 as appropriate, including Malware Subject labels and Capabilities/Objectives.