# Introduction

# Background Information

# Data Models

## MeasureSourceType Class

The MeasureSourceType class is a type representing a description of a single cyber observation source.

Table 3‑1. Properties of the MeasureSourceType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **class** | SourceClassTypeEnum | 0..1 | The class property is optional and enables identification of the high-level class of this cyber observation source. |
| **source\_type** | SourceTypeEnum | 0..1 | The source\_type property is optional and enables identification of the broad type of this cyber observation source. |
| **name** | basicDataTypes:  BasicString | 0..1 | The name property is optional and enables the assignment of a relevant name to this Discovery Method. |
| **sighting\_count** | basicDataTypes:  PositiveInteger | 0..1 | The sighting\_count property specifies how many different identical instances of a given Observable may have been seen/sighted by the observation source. |
| **Information\_Source\_Type** | cyboxCommon:  VocabularyStringType | 0..1 | The Information\_Source\_Type property ??????????. Examples of potential ???? (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is ?????. |
| **Tool\_Type** | cyboxCommon:  VocabularyStringType | 0..1 | The Tool\_Type property ??????????. Examples of potential ???? (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is ?????. |
| **Description** | StructuredTextType | 0..1 | The Description property captures a technical description of the MeasureSource. Any length is permitted. Optional formatting is supported via the structuring\_format property of the StructuredTextType class. |
| **Contributors** | PersonnelType | 0..1 | The Contributors property is optional and enables description of the individual contributors involved in this cyber observation source. |
| **Time** | TimeType | 0..1 | The Time property is optional and enables description of various time-related properties for this cyber observation source instance. |
| **Observation\_Location** | LocationType | 0..1 | The Observation\_Location property specifies a relevant physical location for the observation measurement of the associated Observable. This field is implemented through the xsi:type extension mechanism. The default type is CIQAddressInstanceType in the http://cybox.mitre.org/extensions/Identity#CIQAddress-1 namespace. This type is defined in the extensions/location/ciq\_address\_3.0.xsd file or at the URL http://cybox.mitre.org/XMLSchema/extensions/location/ciq\_address/1.0/ciq\_address\_3.0.xsd. Those who wish to express a simple name may also do so by not specifying an xsi:type and using the Name field. |
| **Tools** | ToolsInformationType | 0..1 | The Tools property is optional and enables description of the tools utilized for this cyber observation source. |
| **Platform** | PlatformSpecificationType | 0..1 | The Platform property is optional and enables a formal, standardized specification of the platform for this cyber observation source. |
| **System** | ObjectPropertiesType | 0..1 | The System property is optional and enables characterization of the system on which the mechanism of cyber observation executed. System should be an object of type SystemObj:SystemObjectType. |
| **Instance** | ObjectPropertiesType | 0..1 | The Instance property is optional and enables characterization of the process instance in which the mechanism of cyber observation executed. Instance should be of type ProcessObj:ProcessObjectType. |
| **Observable\_Location** | LocationType | 0..1 | The Observable\_Location property specifies a relevant physical location for the associated Observable. This field is implemented through the xsi:type extension mechanism. The default type is CIQAddressInstanceType in the http://cybox.mitre.org/extensions/Identity#CIQAddress-1 namespace. This type is defined in the extensions/location/ciq\_address\_3.0.xsd file or at the URL http://cybox.mitre.org/XMLSchema/extensions/location/ciq\_address/1.0/ciq\_address\_3.0.xsd. Those who wish to express a simple name may also do so by not specifying an xsi:type and using the Name field. |

## ContributorType Class

The ContributorType class represents a description of an individual who contributed as a source of cyber observation data.

Table 3‑2. Properties of the ContributorType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Role** | basicDataTypes:  BasicString | 0..1 | The Role property describes the role played by this contributor. |
| **Name** | basicDataTypes:  BasicString | 0..1 | The Name property contains the name of this contributor. |
| **Email** | basicDataTypes:  BasicString | 0..1 | The Email property contains the email of this contributor. |
| **Phone** | basicDataTypes:  BasicString | 0..1 | The Phone property contains a telephone number of this contributor. |
| **Organization** | basicDataTypes:  BasicString | 0..1 | The Organization property contains the organization name of this contributor. |
| **Date** | DateRangeType | 0..1 | The Date property contains a description (bounding) of the timing of this contributor's involvement. |
| **Contribution\_Location** | basicDataTypes:  BasicString | 0..1 | The Contribution\_Location property contains information describing the location at which the contributory activity occurred. |

## DateRangeType Class

The DateRangeType class specifies a range of dates.

Properties of the DateRangeType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Start\_Date** | DateWithPrecisionType | 0..1 | The Start\_Date property contains the start date for this contributor's involvement. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |
| **End\_Date** | DateWithPrecisionType | 0..1 | The End\_date property contains the end date for this contributor's involvement. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |

## PersonnelType Class

The PersonnelType class is an abstracted data type to standardize the description of sets of personnel.

Table 3‑3. Properties of the PersonnelType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Contributor** | ContributorType | 1..\* | This property contains information describing the identify, resources and timing of involvement for a single contributor. |

## TimeType Class

The TimeType class specifies various time properties for this construct.

Table 3‑4. Properties of the TimeType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Start\_Time** | DateTimeWithPrecisionType | 0..1 | The Start\_Time property is optional and describes the starting time for this property. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |
| **End\_Time** | DateTimeWithPrecisionType | 0..1 | The End\_Time property is optional and describes the ending time for this property. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |
| **Produced\_Time** | DateTimeWithPrecisionType | 0..1 | The Produced\_Time property is optional and describes the time that this property was produced. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |
| **Received\_Time** | DateTimeWithPrecisionType | 0..1 | The Received\_Time property is optional and describes the time that this property was received. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |

## ToolSpecificDataType Class

The ToolSpecificDataType class is an Abstract type placeholder within the CybOX schema enabling the inclusion of metadata for a specific type of tool through the use of a custom type defined as an extension of this base Abstract type.

Properties of the ToolSpecificDataType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## ToolsInformationType Class

The ToolsInformationType class represents a description of a set of automated tools.

Properties of the ToolsInformationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Tool** | ToolInformationType | 1..\* | The Tool property is optional and enables description of a single tool utilized for this cyber observation source. |

## ToolInformationType Class

The ToolInformationType class is intended to characterize the properties of a hardware or software tool, including those related to instances of its use.

Table 3‑5. Properties of the ToolInformationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **id** | basicDataTypes:  QualifiedName | 0..1 | The id property specifies a globally unique identifier for the ToolInformation. |
| **idref** | basicDataTypes:  QualifiedName | 0..1 | The idref property specifies an identifier reference to a ToolInformation instance specified elsewhere. When the idref property is used, no other property should be specified. |
| **Name** | basicDataTypes:  BasicString | 0..1 | The Name property contains the name of the tool leveraged. |
| **Type** | cyboxCommon:  VocabularyStringType | 0..\* | The Type property ??????????. Examples of potential ???? (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is ?????. |
| **Description** | StructuredTextType | 0..1 | The Description property captures a technical description of the ToolInformation. Any length is permitted. Optional formatting is supported via the structuring\_format property of the StructuredTextType class. |
| **References** | ToolReferencesType | 0..1 | The References property contains references to instances or additional information for this tool. |
| **Vendor** | basicDataTypes:  BasicString | 0..1 | The Vendor property contains information identifying the vendor organization for this tool. |
| **Version** | basicDataTypes:  BasicString | 0..1 | The Version property contains an appropriate version descriptor of this tool. |
| **Service\_Pack** | basicDataTypes:  BasicString | 0..1 | The Service\_Pack property contains an appropriate service pack descriptor for this tool. |
| **Tool\_Specific\_Data** | ToolSpecificDataType | 0..1 | The Tool\_Specific\_Data property This is an abstract type provided to a flexible mechanism for enabling tool-specific data to be included. |
| **Tool\_Hashes** | HashListType | 0..1 | The Tool\_Hashes property contains a hash value computed on the tool file content in order to verify its integrity. |
| **Tool\_Configuration** | ToolConfigurationType | 0..1 | The Tool\_Configuation property contains information describing the configuration and usage of the tool. |
| **Execution\_Environment** | ExecutionEnvironmentType | 0..1 | The Execution\_Environment property contains information describing the execution environment of the tool. |
| **Errors** | ErrorsType | 0..1 | The Errors property captures any errors generated during the run of the tool. |
| **Metadata** | MetadataType | 0..\* | The Metadata property captures other relevant metadata including tool-specific fields. |
| **Compensation\_Model** | CompensationModelType | 0..1 | The Compensation\_Model property contains the name of the compensation model used for the tool. |

## CompensationModelType Class

The CompensationModelType class characterizes the compensation model for a tool.

Properties of the CompensationModelType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## ToolReferencesType Class

Used to indicate one or more references to tool instances and information.

Table 3‑6. Properties of the ToolReferencesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Reference** | ToolReferenceType | 1..\* | The Reference property Contains one reference to information or instances of a given tool. |

## ToolReferenceType Class

Contains one reference to information or instances of a given tool.

Table 3‑7. Properties of the ToolReferenceType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **reference\_type** | ToolReferenceTypeEnum | 0..1 | The reference\_type property Indicates the nature of the referenced material (documentation, source, executable, etc.). |

## ToolConfigurationType Class

The ToolConfigurationType class characterizes the configuration for a tool used as a cyber observation source.

Table 3‑8. Properties of the ToolConfigurationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Configuration\_Settings** | ConfigurationSettingsType | 0..1 | The Configuration\_Settings property describes the configuration settings of this tool instance. |
| **Dependencies** | DependenciesType | 0..1 | The Dependencies property contains information describing the relevant dependencies for this tool. |
| **Usage\_Context\_Assumptions** | UsageContextAssumptionsType | 0..1 | The Usage\_Context\_Assumptions property contains descriptions of the various relevant usage context assumptions for this tool . |
| **Internationalization\_Settings** | Internationalization  SettingsType | 0..1 | The Internationalization\_Settings property contains information describing relevant internationalization setting for this tool . |
| **Build\_Information** | BuildInformationType | 0..1 | The Build\_information property contains information describing how this tool was built. |

## ConfigurationSettingsType Class

The ConfigurationSettingsType class is a modularized data type used to provide a consistent approach to describing configuration settings for a tool, application or other cyber object.

Table 3‑9. Properties of the ConfigurationSettingsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Configuration\_Setting** | ConfigurationSettingType | 1..\* | The Configuration\_Setting property contains a single configuration setting instance. |

## ConfigurationSettingType Class

The ConfigurationSettingType class is a modularized data type used to provide a consistent approach to describing a particular configuration setting for a tool, application or other cyber object.

Table 3‑10. Properties of the ConfigurationSettingType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Item\_Name** | basicDataTypes:  BasicString | 1 | The Item\_Name property contains the name of the configuration item referenced by this configuration setting instance. |
| **Item\_Value** | basicDataTypes:  BasicString | 1 | The Item\_Value property contains the value of this configuration setting instance. |
| **Item\_Type** | basicDataTypes:  BasicString | 0..1 | The Item\_Type property contains the type of the configuration item referenced in this configuration setting instance. |
| **Item\_Description** | basicDataTypes:  BasicString | 0..1 | The Item\_Description property contains a description of the configuration item referenced in this configuration setting instance. |

## DependenciesType Class

The DependenciesType class contains information describing a set of dependencies for this tool.

Table 3‑11. Properties of the DependenciesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Dependency** | DependencyType | 1..\* | The Dependency property contains information describing a single dependency for this tool. |

## DependencyType Class

The DependencyType class contains information describing a single dependency for this tool.

Table 3‑12. Properties of the DependencyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Dependency\_Type** | basicDataTypes:  BasicString | 0..1 | The Dependency\_Type property describes the type of this dependency instance. |
| **Dependency\_Description** | StructuredTextType | 1 | The Dependency\_Description property contains a description of this dependency instance. |

## UsageContextAssumptionsType Class

The UsageContextAssumptionsType class contains descriptions of the various relevant usage context assumptions for this tool.

Table 3‑13. Properties of the UsageContextAssumptionsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Usage\_Context\_Assumption** | StructuredTextType | 1..\* | The Usage\_Context\_Assumption property contains a single usage context assumption for this tool. |

## InternationalizationSettingsType Class

The InternationalizationSettingsType class contains information describing relevant internationalization setting for this tool.

Table 3‑14. Properties of the InternationalizationSettingsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Internal\_Strings** | InternalStringsType | 1..\* | The Internal\_Strings property contains a single internal string instance for this internationalization setting instance. |

## InternalStringsType Class

The InternalStringsType class contains a single internal string instance for this internationalization setting instance.

Table 3‑15. Properties of the InternalStringsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Key** | basicDataTypes:  BasicString | 1 | The Key property contains the actual key of this internal string instance. |
| **Content** | basicDataTypes:  BasicString | 1 | The Content property contains the actual content of this internal string instance. |

## BuildInformationType Class

The BuildInformationType class contains information describing how this tool was built.

Table 3‑16. Properties of the BuildInformationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Build\_ID** | basicDataTypes:  BasicString | 0..1 | The Build\_ID property contains an externally defined unique identifier of this build of this application instance. |
| **Build\_Project** | basicDataTypes:  BasicString | 0..1 | The Build\_Project property contains the project name of this build of this application instance. |
| **Build\_Utility** | BuildUtilityType | 0..1 | The Build\_Utility property contains information identifying the utility used to build this application. |
| **Build\_Version** | basicDataTypes:  BasicString | 0..1 | The Build\_Version property contains the appropriate version descriptor of this build of this application instance. |
| **Build\_Label** | basicDataTypes:  BasicString | 0..1 | The Build\_Label property contains any relevant label for this build of this application instance. |
| **Compilers** | CompilersType | 0..1 | The Compilers property describes the compilers utilized during this build of this application. |
| **Compilation\_Date** | DateTimeWithPrecisionType | 0..1 | The Completion\_Date property identifies the compilation date for the build of the tool. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |
| **Build\_Configuration** | BuildConfigurationType | 0..1 | The Build\_Configuration property describes how the build utility was configured for this build of this application. |
| **Build\_Script** | basicDataTypes:  BasicString | 0..1 | The Build\_Script property contains the actual build script for this build of this application instance. |
| **Libraries** | LibrariesType | 0..1 | The Libraries property identifies the libraries incorporated into the build of the tool. |
| **Build\_Output\_Log** | basicDataTypes:  BasicString | 0..1 | The Build\_Output\_Log property contains a capture of the output log of the build process. |

## BuildUtilityType Class

The BuildUtilityType class contains information identifying the utility used to build this application.

Table 3‑17. Properties of the BuildUtilityType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Build\_Utility\_Name** | basicDataTypes:  BasicString | 1 | The Build\_Utility\_Name property contains the informally defined name of the utility used to build this application instance. |
| **Build\_Utility\_Platform\_Specification** | PlatformSpecificationType | 1 | The Build\_Utility\_Platform\_Specification property identifies the build utility used to build this application. |

## CompilersType Class

The CompilersType class describes the compilers utilized during this build of this application.

Table 3‑18. Properties of the CompilersType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Compiler** | CompilerType | 1..\* | The Compiler property describes a single compiler utilized during this build of this application. |

## CompilerType Class

The CompilerType class describes a single compiler utilized during this build of this application.

Table 3‑19. Properties of the CompilerType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Compiler\_Informal\_Description** | CompilerInformalDescriptionType | 0..1 | The Compiler\_Informal\_Description property contains the informal description of this compiler instance. |
| **Compiler\_Platform\_Specification** | PlatformSpecificationType | 0..1 | The Compiler\_Platform\_Specification property identifies this compiler instance. |

## CompilerInformalDescriptionType Class

The CompilerInformalDescriptionType class contains the informal description of this compiler instance.

Table 3‑20. Properties of the CompilerInformalDescriptionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Compiler\_Name** | basicDataTypes:  BasicString | 1 | The Compiler\_Name property contains the name of the compiler. |
| **Compiler\_Version** | basicDataTypes:  BasicString | 0..1 | The Compiler\_Version property contains the version of the compiler. |

## BuildConfigurationType Class

The BuildConfigurationType class describes how the build utility was configured for this build of this application.

Table 3‑21. Properties of the BuildConfigurationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Configuration\_Setting\_Description** | basicDataTypes:  BasicString | 0..1 | The Configuration\_Setting\_Description property contains the description of the configuration settings for this build of this application instance. |
| **Configuration\_Settings** | ConfigurationSettingsType | 1 | The Configuration\_Settings property contains the configuration settings for this build of this application instance. |

## LibrariesType Class

The LibrariesType class identifies the libraries incorporated into the build of the tool.

Table 3‑22. Properties of the LibrariesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Library** | LibraryType | 0..1 | The Library property identifies a library incorporated into the build of the tool. |

## LibraryType Class

The LibraryType class identifies a single library incorporated into the build of the tool.

Table 3‑23. Properties of the LibraryType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **name** | basicDataTypes:  BasicString | 0..1 | The name property identifies the name of the library. |
| **version** | basicDataTypes:  BasicString | 0..1 | The version property identifies the version of the library. |

## ExecutionEnvironmentType Class

The ExecutionEnvironmentType class contains information describing the execution environment of the tool.

Table 3‑24. Properties of the ExecutionEnvironmentType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **System** | ObjectPropertiesType | 0..1 | The System property contains information describing the system on which the tool was executed. The system should be of class SystemObj:SystemObjectType. |
| **User\_Account\_Info** | ObjectPropertiesType | 0..1 | The User\_Account\_info property contains information describing the user account that executed the tool. User\_Account\_Info should be of class UserAccountObj:UserAccountObjectType. |
| **Command\_Line** | basicDataTypes:  BasicString | 0..1 | The Command\_Line property specifies the command line string used to run the tool. |
| **Start\_Time** | DateTimeWithPrecisionType | 0..1 | The Start\_Time property specifies when the tool was run. In order to avoid ambiguity, it is strongly suggest that all timestamps in this field include a specification of the timezone if it is known. |

## ErrorsType Class

The ErrorsType class captures any errors generated during the run of the tool.

Table 3‑25. Properties of the ErrorsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Error** | ErrorType | 1..\* | The Error property captures a single type of error generated during the run of the tool. |

## ErrorType Class

The ErrorType class captures a single error generated during the run of the tool.

Table 3‑26. Properties of the ErrorType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Error\_Type** | basicDataTypes:  BasicString | 1 | The Error\_Type property specifies the type for this tool run error. |
| **Error\_Count** | basicDataTypes:  PositiveInteger | 0..1 | The Error\_Count property specifies the count of instances for this error in the tool run. |
| **Error\_Instances** | ErrorInstancesType | 0..1 | The Error\_Instances property captures the actual error output for each instance of this type of error. |

## ErrorInstancesType Class

The ErrorInstancesType class captures the actual error output for each instance of this type of error.

Table 3‑27. Properties of the ErrorInstancesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Error\_Instance** | basicDataTypes:  BasicString | 1..\* | The Error\_Instance property captures the actual error output for a single instance of this type of error. |

## ObjectPropertiesType Class

The ObjectPropertiesType class is an abstract class within the CybOX schema enabling the inclusion of contextually varying object properties descriptions. This Abstract type is leveraged as the extension base for all predefined CybOX object properties schemas. Through this extension mechanism any object instance data based on an object properties schema extended from ObjectPropertiesType (e.g. File\_Object, Address\_Object, etc.) can be directly integrated into any instance document where a field is defined as ObjectPropertiesType. For flexibility and extensibility purposes any user of CybOX can specify their own externally defined object properties schemas (outside of or derived from the set of predefined objects) extended from ObjectPropertiesType class and utilize them as part of their CybOX content.

Table 3‑28. Properties of the ObjectPropertiesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **object\_reference** | basicDataTypes:  QualifiedName | 0..1 | The object\_reference property specifies a unique ID reference to an Object defined elsewhere. This property allows for the re-use of the defined Properties of one Object within another, without the need to embed the full Object in the location from which it is being referenced. Thus, this ID reference is intended to resolve to the Properties of the Object that it points to. |
| **Custom\_Properties** | CustomPropertiesType | 0..1 | The Custom\_Properties property is optional and enables the specification of a set of custom Object Properties that may not be defined in existing Properties schemas. |

## CustomPropertiesType Class

The CustomPropertiesType class enables the specification of a set of custom Object Properties that may not be defined in existing Properties schemas.

Table 3‑29. Properties of the CustomPropertiesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Property** | PropertyType | 1..\* | The Property property enables the specification of a single Object Property. |

## PropertyType Class

The PropertyType class is a type representing the specification of a single Object Property.

Table 3‑30. Properties of the PropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **name** | basicDataTypes:  BasicString | 0..1 | The name property specifies a name for this property. |
| **description** | basicDataTypes:  BasicString | 0..1 | The description property A description of what this property represents. |

## BaseObjectPropertyType Class

The BaseObjectPropertyType class is a type representing a common typing foundation for the specification of a single Object Property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Table 3‑31. Properties of the BaseObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **id** | basicDataTypes:  QualifiedName | 0..1 | The id property specifies a globally unique identifier for the BaseObjectProperty. |
| **idref** | basicDataTypes:  QualifiedName | 0..1 | The idref property specifies an identifier reference to a BaseObjectProperty instance specified elsewhere. When the idref property is used, no other property should be specified. |
| **datatype** | DatatypeEnum | 0..1 | The datatype property This attribute is optional and specifies the expected type for the value of the specified property. |
| **appears\_random** | basicDataTypes:  Boolean | 0..1 | The appear\_random property is optional and conveys whether the associated object property value appears to somewhat random in nature. An object property with this field set to TRUE need not provide any further information including a value. If more is known about the particular variation of randomness, a regex value could be provided to outline what is known of the structure. |
| **is\_obfuscated** | basicDataTypes:  Boolean | 0..1 | The is\_obfuscated property is optional and conveys whether the associated Object property has been obfuscated. |
| **obfuscation\_algorithm\_ref** | basicDataType:URI | 0..1 | The obfuscation\_algorithm\_ref property is optional and conveys a reference to a description of the algorithm used to obfuscate this Object property. |
| **is\_defanged** | basicDataTypes:  Boolean | 0..1 | The is\_defanged property is optional and conveys whether the associated Object property has been defanged (representation changed to prevent malicious effects of handling/processing). |
| **defanging\_algorithm\_ref** | basicDataType:URI | 0..1 | The defanging\_algorithm\_ref property is optional and conveys a reference to a description of the algorithm used to defang (representation changed to prevent malicious effects of handling/processing) this Object property. |
| **refanging\_transform\_type** | basicDataTypes:  BasicString | 0..1 | The refanging\_transform\_type property is optional and specifies the type (e.g. RegEx) of refanging transform specified in the optional accompanying refangingTransform property. |
| **refanging\_transform** | basicDataTypes:  BasicString | 0..1 | The refanging\_transform property is optional and specifies an automated transform that can be applied to the Object property content in order to refang it to its original format. |
| **observed\_encoding** | basicDataTypes:  BasicString | 0..1 | The observed\_encoding property is optional and specifies the encoding of the string when it is/was observed. This may be different from the encoding used to represent the string within this element. It is strongly recommended that character set names should be taken from the IANA character set registry (https://www.iana.org/assignments/character-sets/character-sets.xhtml). This field is intended to be applicable only to fields which contain string values. |
| **condition** | ConditionTypeEnum | 0..1 | The condition property is optional and defines the relevant condition to apply to the value. |
| **is\_case\_sensitive** | basicDataTypes:  Boolean | 0..1 | The is\_case\_sensitive property is optional and should be used when specifying the case-sensitivity of a pattern which uses an Equals, DoesNotEqual, Contains, DoesNotContain, StartsWith, EndsWith, or FitsPattern condition. The default value for this field is "true" which indicates that pattern evaluations are to be considered case-sensitive. |
| **apply\_condition** | ConditionApplicationEnum | 0..1 | The apply\_condition property indicates how a condition should be applied when the field body contains a list of values. (Its value is moot if the field value contains only a single value - both possible values for this field would have the same behavior.) If this field is set to ANY, then a pattern is considered to be matched if the provided condition successfully evaluates for any of the values in the field body. If the field is set to ALL, then the patern only matches if the provided condition successfully evaluates for every value in the field body. |
| **delimiter** | basicDataTypes:  BasicString | 0..1 | The delimiter property specifies the delimiter used when defining lists of values. The default value is "##comma##". |
| **bit\_mask** | basicDataType:  HexBinary | 0..1 | The bit\_mask property Used to specify a bit\_mask in conjunction with one of the defined binary conditions (bitwiseAnd, bitwiseOr, and bitwiseXor). This bitmask is then uses as one operand in the indicated bitwise computation. |
| **pattern\_type** | PatternTypeEnum | 0..1 | The pattern\_type property is optional and defines the type of pattern used if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. |
| **regex\_syntax** | basicDataTypes:  BasicString | 0..1 | The regex\_syntax property is optional and defines the syntax format used for a regular expression, if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. Setting this attribute with an empty value (e.g., "") or omitting it entirely notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities, character classes, escapes, and other lexical tokens defined by the CybOX Language Specification. Setting this attribute with a non-empty value notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities not defined by the CybOX Language Specification. The regular expression must be evaluated through a compatible regular expression engine in this case. |
| **has\_changed** | basicDataTypes:  Boolean | 0..1 | The has\_changed property is optional and conveys a targeted observation pattern of whether the associated field value has changed. This field would be leveraged within a pattern observable triggering on whether the value of a single field value has changed. |
| **trend** | basicDataTypes:  Boolean | 0..1 | The trend property is optional and conveys a targeted observation pattern of the nature of any trend in the associated field value. This field would be leveraged within a pattern observable triggering on the matching of a specified trend in the value of a single specified field. |

## IntegerObjectPropertyType Class

The IntegerObjectPropertyType class is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Int. This type will be assigned to any property of a CybOX object that should contain content of type Integer and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the IntegerObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## StringObjectPropertyType Class

The StringObjectPropertyType class is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type String. This type will be assigned to any property of a CybOX object that should contain content of type String and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the StringObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## NameObjectPropertyType Class

The NameObjectPropertyType class is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Name. This type will be assigned to any property of a CybOX object that should contain content of type Name and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the NameObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## DateObjectPropertyRestrictionType Class

The DateObjectPropertyRestrictionType class is a type is an intermediate type to allow for the addition of the precision attribute to DateObjectPropertyType. It should not be used directly.

Properties of the DateObjectPropertyRestrictionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## DateObjectPropertyType Class

The DateObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Date. This type will be assigned to any property of a CybOX object that should contain content of type Date and enables the use of relevant metadata for the property. In order to avoid ambiguity, it is strongly suggested that any date representation in this field include a timezone if it is known. As with the rest of the field, this should be formatted per the xs:date specification. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space. For fields of this type using CybOX patterning, it is strongly suggested that the condition (pattern type) is limited to one of Equals, DoesNotEqual, GreaterThan, LessThan, GreaterThanOrEqual, LessThanOrEqual, ExclusiveBetween, or InclusiveBetween. The use of other conditions may lead to ambiguity or unexpected results. When evaluating data against a pattern, the evaluator should take into account the precision of the field (as given by the precision attribute) and any timezone information that is available to perform a data-aware comparison. The usage of simple string comparisons is discouraged due to ambiguities in how precision and timezone information is processed.

Table 3‑32. Properties of the DateObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **precision** | DatePrecisionEnum | 0..1 | The precision property of the associated time. If omitted, the default is "day", meaning the full property value. Digits in the date that are required by the xs:date datatype but are beyond the specified precision should be zeroed out. When used in conjunction with CybOX patterning, the pattern should only be evaluated against the target up to the given precision. |

## DateTimeObjectPropertyRestrictionType Class

The DateTimeObjectPropertyRestrictionType class is type is an intermediate type to allow for the addition of the precision attribute to DateTimeObjectPropertyType. It should not be used directly.

Properties of the DateTimeObjectPropertyRestrictionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## DateTimeObjectPropertyType Class

The DateTimeObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type DateTime. This type will be assigned to any property of a CybOX object that should contain content of type DateTime and enables the use of relevant metadata for the property. In order to avoid ambiguity, it is strongly suggested that any dateTime representation in this field include a timezone. As with the rest of the field, this should be formatted per the xs:dateTime specification. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space. For fields of this type using CybOX patterning, it is strongly suggested that the condition (pattern type) is limited to one of Equals, DoesNotEqual, GreaterThan, LessThan, GreaterThanOrEqual, LessThanOrEqual, ExclusiveBetween, or InclusiveBetween. The use of other conditions may lead to ambiguity or unexpected results. When evaluating data against a pattern, the evaluator should take into account the precision of the field (as given by the precision attribute) and any timezone information that is available to perform a data-aware comparison. The usage of simple string comparisons is discouraged due to ambiguities in how precision and timezone information is processed.

Table 3‑33. Properties of the DateTimeObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Precision** | DateTimePrecisionEnum | 0..1 | The precision property of the associated time. If omitted, the default is "second", meaning the full property value (including fractional seconds). Digits in the dateTime that are required by the xs:dateTime datatype but are beyond the specified precision should be zeroed out. When used in conjunction with CybOX patterning, the pattern should only be evaluated against the target up to the given precision. |

## FloatObjectPropertyType Class

The FloatObjectPropertyType class is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Float. This type will be assigned to any property of a CybOX object that should contain content of type Float and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the FloatObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## DoubleObjectPropertyType Class

The DoubleObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Double. This type will be assigned to any property of a CybOX object that should contain content of type Double and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the DoubleObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## UnsignedLongObjectPropertyType Class

The UnsignedLongObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type UnsignedLong. This type will be assigned to any property of a CybOX object that should contain content of type UnsignedLong and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the UnsignedLongObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## UnsignedIntegerObjectPropertyType Class

The UnsignedIntegerObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type UnsignedInt. This type will be assigned to any property of a CybOX object that should contain content of type UnsignedInteger and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the UnsignedIntegerObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## PositiveIntegerObjectPropertyType Class

The PositiveIntegerObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type PositveInteger. This type will be assigned to any property of a CybOX object that should contain content of type PositiveInteger and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the PositiveIntegerObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## HexBinaryObjectPropertyType Class

The HexBinaryObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type HexBinary. This type will be assigned to any property of a CybOX object that should contain content of type HexBinary and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the HexBinaryObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## LongObjectPropertyType Class

The LongObjectPropertyType class is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type Long. This type will be assigned to any property of a CybOX object that should contain content of type Long and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the LongObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## NonNegativeIntegerObjectPropertyType Class

The NonNegativeIntegerObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type nonNegativeInteger. This type will be assigned to any property of a CybOX object that should contain content of type NonNegativeInteger and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the NonNegativeIntegerObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## AnyURIObjectPropertyType Class

The AnyURIObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type anyURI. This type will be assigned to any property of a CybOX object that should contain content of type AnyURI and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the AnyURIObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## DurationObjectPropertyType Class

The DurationObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type duration. This type will be assigned to any property of a CybOX object that should contain content of type Duration and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the DurationObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## TimeObjectPropertyRestrictionType Class

The DateTimeObjectPropertyRestrictionType class is a type is an intermediate type to allow for the addition of the precision attribute to TimeObjectPropertyType. It should not be used directly.

Properties of the TimeObjectPropertyRestrictionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## TimeObjectPropertyType Class

The TimeObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type time. This type will be assigned to any property of a CybOX object that should contain content of type Time and enables the use of relevant metadata for the property. In order to avoid ambiguity, it is strongly suggested that any time representation in this field include a specification of the timezone if it is known. As with the rest of the field, this should be formatted per the xs:time specification. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space. For fields of this type using CybOX patterning, it is strongly suggested that the condition (pattern type) is limited to one of Equals, DoesNotEqual, GreaterThan, LessThan, GreaterThanOrEqual, LessThanOrEqual, ExclusiveBetween, or InclusiveBetween. The use of other conditions may lead to ambiguity or unexpected results. When evaluating data against a pattern, the evaluator should take into account the precision of the field (as given by the precision attribute) and any timezone information that is available to perform a data-aware comparison. The usage of simple string comparisons is discouraged due to ambiguities in how precision and timezone information is processed.

Table 3‑34. Properties of the TimeObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **precision** | TimePrecisionEnum | 0..1 | The precision property specifies the granularity with which a timestamp should be considered as specified by the TypePrecsionEnum enumeration (e.g., '*hour*,' '*minute*'). If omitted, the default precision is '*second*.' Digits in a timestamp that are beyond a specified precision SHOULD be zeroed out.When used in conjuction with CybOX patterning, the pattern should only be evaluated against the target up to the given precision |

## Base64BinaryObjectPropertyType Class

The Base64BinaryObjectPropertyType is a type (extended from BaseObjectPropertyType) representing the specification of a single Object property whose core value is of type base64Binary. This type will be assigned to any property of a CybOX object that should contain content of type Base64Binary and enables the use of relevant metadata for the property. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the Base64BinaryObjectPropertyType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## BaseObjectPropertyGroup Class

The ObjectPropertyGroup is a simple field group aggregating a set of fields for Object Properties.

Properties of the BaseObjectPropertyGroup class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **id** | basicDataTypes:  QualifiedName | 0..1 | The id property specifies a globally unique identifier for the BaseObjectPropertyGroup. |
| **idref** | basicDataTypes:  QualifiedName | 0..1 | The idref property specifies an identifier reference to a BaseObjectPropertyGroup instance specified elsewhere. When the idref property is used, no other property should be specified. |
| **datatype** | DatatypeEnum | 0..1 | The datatype property This attribute is optional and specifies the expected type for the value of the specified property. |
| **appears\_random** | basicDataTypes:  Boolean | 0..1 | This property is optional and conveys whether the associated object property value appears to somewhat random in nature. An object property with this field set to TRUE need not provide any further information including a value. If more is known about the particular variation of randomness, a regex value could be provided to outline what is known of the structure. |
| **is\_obfuscated** | basicDataTypes:  Boolean | 0..1 | This property is optional and conveys whether the associated Object property has been obfuscated. |
| **obfuscation\_algorithm\_ref** | xs:anyURI | 0..1 | This property is optional and conveys a reference to a description of the algorithm used to obfuscate this Object property. |
| **is\_defanged** | basicDataTypes:  Boolean | 0..1 | This property is optional and conveys whether the associated Object property has been defanged (representation changed to prevent malicious effects of handling/processing). |
| **defanging\_algorithm\_ref** | xs:anyURI | 0..1 | This property is optional and conveys a reference to a description of the algorithm used to defang (representation changed to prevent malicious effects of handling/processing) this Object property. |
| **refanging\_transform\_type** | basicDataTypes:BasicString | 0..1 | This property is optional and specifies the type (e.g. RegEx) of refanging transform specified in the optional accompanying refangingTransform property. |
| **refanging\_transform** | basicDataTypes:BasicString | 0..1 | This property is optional and specifies an automated transform that can be applied to the Object property content in order to refang it to its original format. |
| **observed\_encoding** | basicDataTypes:BasicString | 0..1 | This property is optional and specifies the encoding of the string when it is/was observed. This may be different from the encoding used to represent the string within this element. It is strongly recommended that character set names should be taken from the IANA character set registry (https://www.iana.org/assignments/character-sets/character-sets.xhtml). This field is intended to be applicable only to fields which contain string values. |

## PatternFieldGroup Class

The PatternFieldGroup is a simple field group aggregating a set of fields for application of patterns.

Properties of the PatternFieldGroup class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **condition** | ConditionTypeEnum | 0..1 | This property is optional and defines the relevant condition to apply to the value. |
| **is\_case\_sensitive** | basicDataTypes:Boolean | 0..1 | The is\_case\_sensitive property is optional and should be used when specifying the case-sensitivity of a pattern which uses an Equals, DoesNotEqual, Contains, DoesNotContain, StartsWith, EndsWith, or FitsPattern condition. The default value for this field is "true" which indicates that pattern evaluations are to be considered case-sensitive. |
| **apply\_condition** | ConditionApplicationEnum | 0..1 | This property indicates how a condition should be applied when the field body contains a list of values. (Its value is moot if the field value contains only a single value - both possible values for this field would have the same behavior.) If this field is set to ANY, then a pattern is considered to be matched if the provided condition successfully evaluates for any of the values in the field body. If the field is set to ALL, then the patern only matches if the provided condition successfully evaluates for every value in the field body. |
| **delimiter** | basicDataTypes:BasicString | 0..1 | The delimiter property specifies the delimiter used when defining lists of values. The default value is "##comma##". |
| **bit\_mask** | xs:hexBinary | 0..1 | The bit\_mask property Used to specify a bit\_mask in conjunction with one of the defined binary conditions (bitwiseAnd, bitwiseOr, and bitwiseXor). This bitmask is then uses as one operand in the indicated bitwise computation. |
| **pattern\_type** | PatternTypeEnum | 0..1 | This property is optional and defines the type of pattern used if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. |
| **regex\_syntax** | basicDataTypes:BasicString | 0..1 | This property is optional and defines the syntax format used for a regular expression, if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. Setting this attribute with an empty value (e.g., "") or omitting it entirely notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities, character classes, escapes, and other lexical tokens defined by the CybOX Language Specification. Setting this attribute with a non-empty value notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities not defined by the CybOX Language Specification. The regular expression must be evaluated through a compatible regular expression engine in this case. |
| **has\_changed** | basicDataTypes:Boolean | 0..1 | This property is optional and conveys a targeted observation pattern of whether the associated field value has changed. This field would be leveraged within a pattern observable triggering on whether the value of a single field value has changed. |
| **trend** | basicDataTypes:Boolean | 0..1 | This property is optional and conveys a targeted observation pattern of the nature of any trend in the associated field value. This field would be leveraged within a pattern observable triggering on the matching of a specified trend in the value of a single specified field. |

## LocationType Class

The LocationType class is used to express geographic location information. This type is extended through the xsi:type mechanism. The default type is CIQAddress3.0InstanceType in the http://cybox.mitre.org/extensions/Address#CIQAddress3.0-1 namespace. This type is defined in the extensions/location/ciq\_address\_3.0.xsd file or at the URL http://cybox.mitre.org/XMLSchema/extensions/location/ciq\_address\_3.0/1.0/ciq\_address\_3.0.xsd. Those who wish to express a simple name may also do so by not specifying an xsi:type and using the Name field of this type.

Table 3‑35. Properties of the LocationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **id** | basicDataTypes:  QualifiedName | 0..1 | The id property specifies a globally unique identifier for the Location. |
| **idref** | basicDataTypes:  QualifiedName | 0..1 | The idref property specifies an identifier reference to a Location instance specified elsewhere. When the idref property is used, no other property should be specified. |
| **Name** | basicDataTypes:  BasicString | 0..1 | The Name property allows for expression of an location through a simple name. |

## ExtractedFeaturesType Class

The ExtractedFeaturesType class is a type representing a description of features extracted from an object such as a file.

Table 3‑36. Properties of the ExtractedFeaturesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Strings** | ExtractedStringsType | 0..1 | The Strings property enables description of a set of static strings extracted from a raw cyber object. |
| **Imports** | ImportsType | 0..1 | The Imports property enables description of a set of references to external resources imported by a raw cyber object. |
| **Functions** | FunctionsType | 0..1 | The Functions property enables description of a set of references to functions called by a raw cyber object. |
| **Code\_Snippets** | CodeSnippetsType | 0..1 | The Code\_Snippets property enables description of a set of code snippets extracted from a raw cyber object. |

## ExtractedStringsType Class

The ExtractedStringsType class is intended as container for strings extracted from CybOX objects.

Table 3‑37. Properties of the ExtractedStringsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **String** | ExtractedStringType | 1..\* | The String property enables description of a single static string extracted from a raw cyber object. |

## ExtractedStringType Class

The ExtractedStringType class is intended as container a single string extracted from a CybOX object.

Table 3‑38. Properties of the ExtractedStringType class

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | | **Multiplicity** | **Description** |
| **Encoding** | | cyboxCommon:VocabularyStringType | 0..1 | The Encoding property ??????????. Examples of potential ???? (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is ?????. |
| **String\_Value** | | StringObjectPropertyType | 0..1 | The String\_Value property specifies the actual value of the string extracted from the CybOX object, if it is capable of being represented in the encoding scheme used in the document (most commonly UTF-8). |
| **Byte\_String\_Value** | | HexBinaryObjectPropertyType | 0..1 | The Byte\_String\_Value property specifies the raw, byte-string representation of the string extracted from the CybOX object, in hexadecimal format. |
| **Hashes** | | HashListType | 0..1 | The Hashes property is used to include any hash values computed using the string extracted from the CybOX object as input. |
| **Address** | | HexBinaryObjectPropertyType | 0..1 | The Address property specifies the location or offset of the specified string in the CybOX objects. |
| **Length** | | PositiveIntegerObjectPropertyType | 0..1 | The Length property specifies the length, in characters, of the string extracted from the CybOX object. |
| **Language** | | StringObjectPropertyType | 0..1 | The Language property specifies the language the string is written in, e.g. English. For consistency, we strongly recommend using the ISO 639-2 language code, if available. Please see http://www.loc.gov/standards/iso639-2/php/code\_list.php for a list of ISO 639-2 codes. |
| **English\_Translation** | | StringObjectPropertyType | 0..1 | The English\_Translation property specifies the English translation of the string, if it is not written in English. |

## ImportsType Class

The ImportsType class is intended to represent an extracted list of imports specified within a CybOX object.

Table 3‑39. Properties of the ImportsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Import** | StringObjectPropertyType | 1..\* | The Import property enables description of a single reference to an external resource imported by a raw cyber object. |

## FunctionsType Class

The FunctionsType class is intended to represent an extracted list of functions leveraged within a CybOX object.

Table 3‑40. Properties of the FunctionsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Function** | StringObjectPropertyType | 1..\* | The Function property enables description of a single reference to a function called by a raw cyber object. |

## CodeSnippetsType Class

The CodeSnippetsType class is intended to represent a set of code snippets extracted from within a CybOX object.

Table 3‑41. Properties of the CodeSnippetsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Code\_Snippet** | ObjectPropertiesType | 1..\* | The Code\_Snippet property enables description of a single code snippet extracted from a raw cyber object. Code\_Snippet should be of CodeObj:CodeObjectType. |

## ByteRunsType Class

The ByteRunsType class is used for representing a list of byte runs from within a raw object.

Table 3‑42. Properties of the ByteRunsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Byte\_Run** | ByteRunType | 1..\* | The Byte\_Run property contains a single byte run from the raw object. |

## ByteRunType Class

The ByteRunType class is used for representing a single byte run from within a raw object.

Table 3‑43. Properties of the ByteRunType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Offset** | IntegerObjectPropertyType | 0..1 | The Offset property specifies the offset of the beginning of the byte run as measured from the beginning of the object. |
| **Byte\_Order** | EndiannessType | 0..1 | The Byte\_Order property specifies the endianness of the unpacked (e.g., unencoded, unencrypted, etc.) data contained within the Byte\_Run\_Data field. |
| **File\_System\_Offset** | IntegerObjectPropertyType | 0..1 | The File\_System\_Offset property is relevant only for byte runs of files in forensic analysis.It specifies the offset of the beginning of the byte run as measured from the beginning of the relevant file system. |
| **Image\_Offset** | IntegerObjectPropertyType | 0..1 | The Image\_Offset property is provided for forensic analysis purposes and specifies the offset of the beginning of the byte run as measured from the beginning of the relevant forensic image. |
| **Length** | IntegerObjectPropertyType | 0..1 | The Length property specifies the number of bytes in the byte run. |
| **Hashes** | HashListType | 0..1 | The Hashes property contains computed hash values for this the data in this byte run. |
| **Byte\_Run\_Data** |  | 0..1 | The Byte\_Run\_Data property contains a raw dump of the byte run data, typically enclosed within an XML CDATA section. |

## HashListType Class

The HashListType class is used for representing a list of hash values.

Table 3‑44. Properties of the HashListType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Hash** | HashType | 1..\* | The Hash property specifies a single calculated hash value. |

## HashValueType Class

The HashValueType class is used for specifying the resulting value from a hash calculation.

Table 3‑45. Properties of the HashValueType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Simple\_Hash\_Value** | SimpleHashValueType | 0..1 | The Simple\_Hash\_Value property specifies a single result value of a basic cryptograhic hash function outputting a single hexbinary hash value. |
| **Fuzzy\_Hash\_Value** | FuzzyHashValueType | 0..1 | The Fuzzy\_Hash\_Value property specifies a single result value of a cryptograhic fuzzy hash function outputting a single complex string based hash value. (e.g. SSDEEP's Block1hash:Block2hash format). |

## SimpleHashValueType Class

The SimpleHashValueType class is used for characterizing the output of basic cryptograhic hash functions outputting a single hexbinary hash value.

Properties of the SimpleHashValueType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## FuzzyHashValueType Class

The FuzzyHashValueType class is used for characterizing the output of cryptograhic fuzzy hash functions outputting a single complex string based hash value.

Properties of the FuzzyHashValueType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## FuzzyHashStructureType Class

The FuzzyHashStructureType class is used for characterizing the internal components of a cryptograhic fuzzy hash algorithmic calculation.

Table 3‑46. Properties of the FuzzyHashStructureType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Block\_Size** | IntegerObjectPropertyType | 0..1 | The Block\_Size property is optional and specifies the calculated block size for this fuzzy hash calculation. |
| **Block\_Hash** | FuzzyHashBlockType | 0..1 | The Block\_Hash property is optional and enables specification of the elemental components utilized for a fuzzy hash calculation on the hashed object utilizing Block\_Size to calculate trigger points. |

## FuzzyHashBlockType Class

The FuzzyHashBlockType class is used for characterizing the internal components of a single block in a cryptograhic fuzzy hash algorithmic calculation.

Table 3‑47. Properties of the FuzzyHashBlockType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Block\_Hash\_Value** | HashValueType | 0..1 | The Block\_Hash\_Value property is optional and specifies a fuzzy hash calculation result value for this Block. |
| **Segment\_Count** | IntegerObjectPropertyType | 0..1 | The Segment\_Count property is optional and specifies the number of segments identified and utilized within this fuzzy hash calculation. |
| **Segments** | HashSegmentsType | 0..1 | The Segments property is optional and specifies the set of segments identified and utilized within this fuzzy hash calculation. |

## HashSegmentsType Class

The HashSegmentsType class is used for characterizing the internal components of a set of trigger point-delimited segments in a cryptographic fuzzy hash algorithmic calculation.

Table 3‑48. Properties of the HashSegmentsType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Segment** | HashSegmentType | 1..\* | The Segment property is optional and specifies a single segment identified and utilized within this fuzzy hash calculation. |

## HashSegmentType Class

The HashSegmentType class is used for characterizing the internal components of a single trigger point-delimited segment in a cryptograhic fuzzy hash algorithmic calculation.

Table 3‑49. Properties of the HashSegmentType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Trigger\_Point** | HexBinaryObjectPropertyType | 0..1 | The Trigger\_Point property is optional and specifies the offset within the hashed object of the trigger point for this segment. |
| **Segment\_Hash** | HashValueType | 0..1 | The Segment\_Hash property is optional and specifies a calculated hash value for this segment. |
| **Raw\_Segment\_Content** |  | 0..1 | The Raw\_Segment\_Content property is optional and contains the raw content of this segment of the hashed object. |

## HashType Class

The HashType class is intended to characterize hash values.

Table 3‑50. Properties of the HashType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Type** | cyboxCommon:  VocabularyStringType | 0..1 | The Type property ??????????. Examples of potential ???? (these specific values are only provided to help explain the property: they are neither recommended values nor necessarily part of any existing vocabulary). The content creator may choose any arbitrary value or may constrain the set of possible values by referencing an externally-defined vocabulary or leveraging a formally defined vocabulary extending from the stixCommon:ControlledVocabularyStringType class. The STIX default vocabulary class for use in the property is ?????. |
| **Simple\_Hash\_Value** | SimpleHashValueType | 0..1 | The Simple\_Hash\_Value property specifies a single result value of a basic cryptograhic hash function outputting a single hexbinary hash value. |
| **Fuzzy\_Hash\_Value** | FuzzyHashValueType | 0..1 | The Fuzzy\_Hash\_Value property specifies a single result value of a cryptograhic fuzzy hash function outputting a single complex string based hash value. (e.g. SSDEEP's Block1hash:Block2hash format). |
| **Fuzzy\_Hash\_Structure** | FuzzyHashStructureType | 0..\* | The Fuzzy\_Hash\_Structure property is optional and enables the characterization of the key internal components of a fuzzy hash calculation with a given block size. |

## StructuredTextType Class

The StructuredTextType class is a type representing a generalized structure for capturing structured or unstructured textual information such as descriptions of things.

Table 3‑51. Properties of the StructuredTextType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **structuring\_format** | basicDataTypes:  BasicString | 0..1 | The structuring\_format property Used to indicate a particular structuring format (e.g., HTML5) used within an instance of StructuredTextType. Note that if the markup tags used by this format would be interpreted as XML information (such as the bracket-based tags of HTML) the text area should be enclosed in a CDATA section to prevent the markup from interfering with XML validation of the CybOX document. If this attribute is absent, the implication is that no markup is being used. |

## DataSegmentType Class

The DataSegmentType is intended to provide a relatively abstract way of characterizing data segments that may be written/read/transmitted or otherwise utilized in actions or behaviors.

Table 3‑52. Properties of the DataSegmentType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **id** | basicDataTypes:  QualifiedName | 0..1 | The id property specifies a globally unique identifier for the DataSegment. |
| **Data\_Format** | DataFormatEnum | 0..1 | The Data\_Format property refers to the type of data contained in the Data\_Segment element. |
| **Data\_Size** | DataSizeType | 0..1 | The Data\_Size property contains the size of the data contained in this element. |
| **Byte\_Order** | EndiannessType | 0..1 | The Byte\_Order property specifies the endianness of the unpacked (e.g., decoded, unencrypted, etc.) data stored within the Data\_Segment field. |
| **Data\_Segment** | StringObjectPropertyType | 0..1 | The Data\_Segment property contains the actual segment of data being characterized. |
| **Offset** | IntegerObjectPropertyType | 0..1 | The Offset property allows for the specification of where to start searching for the specified data segment in an object, in bytes. |
| **Search\_Distance** | IntegerObjectPropertyType | 0..1 | The Search\_Distance property specifies how far into an object should be ignored, in bytes, before starting to search for the specified data segment relative to the end of the previous data segment. |
| **Search\_Within** | IntegerObjectPropertyType | 0..1 | The Search\_Within property specifies that at most N bytes are between data segments in related objects. |

## DataSizeType Class

The DataSizeType class specifies the size of the data segment.

Table 3‑53. Properties of the DataSizeType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **units** | DataSizeUnitsEnum | 0..1 | The units property represents the Units used in the object size element. |

## PlatformSpecificationType Class

The PlatformSpecificationType class is a modularized data type intended for providing a consistent approach to uniquely specifying the identity of a specific platform. In addition to capturing basic information, this type is intended to be extended to enable the structured description of a platform instance using the XML Schema extension feature. The CybOX default extension uses the Common Platform Enumeration (CPE) Applicability Language schema to do so. The extension that defines this is captured in the CPE23PlatformSpecificationType in the http://cybox.mitre.org/extensions/platform#CPE2.3-1 namespace. This type is defined in the extensions/platform/cpe2.3.xsd file.

Table 3‑54. Properties of the PlatformSpecificationType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Description** | StructuredTextType | 0..1 | The Description property captures a technical description of the PlatformSpecification. Any length is permitted. Optional formatting is supported via the structuring\_format property of the StructuredTextType class. |
| **Identifier** | PlatformIdentifierType | 0..\* | The Identifier property Indicates a pre-defined name for the given platform using some naming scheme. For example, one could provide a CPE (Common Platform Enumeration) name using the CPE naming format. |

## PlatformIdentifierType Class

Used to specify a name for a platform using a particular naming system and also allowing a reference pointing to more information about that naming scheme. For example, one could provide a CPE (Common Platform Enumeration) name using the CPE naming format. In this case, the system value could be "CPE" while the system\_ref value could be "http://scap.nist.gov/specifications/cpe/".

Table 3‑55. Properties of the PlatformIdentifierType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **system** | basicDataTypes:  BasicString | 0..1 | The system property Indicates the naming system from which the indicated name was drawn. |
| **system-ref** | basicDataTypes:  URI | 0..1 | The system-ref property A reference to information about the naming system from which the indicated name was drawn. |

## MetadataType Class

The MetadataType class is intended as mechanism to capture any non-context-specific metadata.

Table 3‑56. Properties of the MetadataType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **type** | basicDataTypes:  BasicString | 0..1 | The type property specifies the type of name of a single metadata field. |
| **Value** | basicDataTypes:  BasicString | 0..1 | The Value property specifies the value of name of a single metadata field. |
| **SubDatum** | MetadataType | 0..\* | The SubDatum property uses recursion of the MetadataType specify subdatum structures for this metadata field. |

## EnvironmentVariableListType Class

The EnvironmentVariableListType class is used for representing a list of environment variables.

Table 3‑57. Properties of the EnvironmentVariableListType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Environment\_Variable** | EnvironmentVariableType | 1..\* | The Environment\_Variable property is used for representing environment variables using a name/value pair. |

## EnvironmentVariableType Class

The EnvironmentVariableType class is used for representing environment variables using a name/value pair.

Table 3‑58. Properties of the EnvironmentVariableType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Name** | StringObjectPropertyType | 1 | The Name property specifies the name of the environment variable. |
| **Value** | StringObjectPropertyType | 0..1 | The Value property specifies the value of the environment variable. |

## DigitalSignaturesType Class

The DigitalSignaturesType class is used for representing a list of digital signatures.

Table 3‑59. Properties of the DigitalSignaturesType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **Digital\_Signature** | DigitalSignatureInfoType | 0..\* | The Digital\_Signature property is optional and captures a single digital signature for this Object. |

## DigitalSignatureInfoType Class

The DigitalSignatureInfoType class is used as a way to represent some of the basic information about a digital signature.

Properties of the DigitalSignatureInfoType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **signature\_exists** | basicDataTypes:  Boolean | 0..1 | The signature\_exists property specifies whether the digital signature exists. |
| **signature\_verified** | basicDataTypes:  Boolean | 0..1 | The signature\_verified property specifies if the digital signature is verified. |
| **Certificate\_Issuer** | StringObjectPropertyType | 0..1 | The Certificate\_Issuer property The certificate issuer of the digital signature. |
| **Certificate\_Subject** | StringObjectPropertyType | 0..1 | The Certificate\_Subject property the certificate subject of the digital signature. |
| **Signature\_Description** | StringObjectPropertyType | 0..1 | The Signature\_Description property specifies a description of the digital signature. |

## PatternableFieldType Class

The PatternableFieldType class is a grouping of attributes applicable to defining patterns on a specific field.

Properties of the PatternableFieldType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **condition** | ConditionTypeEnum | 0..1 | This property is optional and defines the relevant condition to apply to the value. |
| **is\_case\_sensitive** | basicDataTypes:Boolean | 0..1 | The is\_case\_sensitive property is optional and should be used when specifying the case-sensitivity of a pattern which uses an Equals, DoesNotEqual, Contains, DoesNotContain, StartsWith, EndsWith, or FitsPattern condition. The default value for this field is "true" which indicates that pattern evaluations are to be considered case-sensitive. |
| **apply\_condition** | ConditionApplicationEnum | 0..1 | This property indicates how a condition should be applied when the field body contains a list of values. (Its value is moot if the field value contains only a single value - both possible values for this field would have the same behavior.) If this field is set to ANY, then a pattern is considered to be matched if the provided condition successfully evaluates for any of the values in the field body. If the field is set to ALL, then the patern only matches if the provided condition successfully evaluates for every value in the field body. |
| **delimiter** | basicDataTypes:BasicString | 0..1 | The delimiter property specifies the delimiter used when defining lists of values. The default value is "##comma##". |
| **bit\_mask** | xs:hexBinary | 0..1 | The bit\_mask property Used to specify a bit\_mask in conjunction with one of the defined binary conditions (bitwiseAnd, bitwiseOr, and bitwiseXor). This bitmask is then uses as one operand in the indicated bitwise computation. |
| **pattern\_type** | PatternTypeEnum | 0..1 | This property is optional and defines the type of pattern used if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. |
| **regex\_syntax** | basicDataTypes:BasicString | 0..1 | This property is optional and defines the syntax format used for a regular expression, if one is specified for the field value. This is applicable only if the Condition field is set to 'FitsPattern'. Setting this attribute with an empty value (e.g., "") or omitting it entirely notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities, character classes, escapes, and other lexical tokens defined by the CybOX Language Specification. Setting this attribute with a non-empty value notifies CybOX consumers and pattern evaluators that the corresponding regular expression utilizes capabilities not defined by the CybOX Language Specification. The regular expression must be evaluated through a compatible regular expression engine in this case. |
| **has\_changed** | basicDataTypes:Boolean | 0..1 | This property is optional and conveys a targeted observation pattern of whether the associated field value has changed. This field would be leveraged within a pattern observable triggering on whether the value of a single field value has changed. |
| **trend** | basicDataTypes:Boolean | 0..1 | This property is optional and conveys a targeted observation pattern of the nature of any trend in the associated field value. This field would be leveraged within a pattern observable triggering on the matching of a specified trend in the value of a single specified field. |

## ControlledVocabularyStringType Class

The ControlledVocabularyStringType is used as the basis for defining controlled vocabularies.

Properties of the ControlledVocabularyStringType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **vocab\_name** | basicDataTypes:  BasicString | 0..1 | The vocab\_name property specifies the name of the controlled vocabulary. |
| **vocab\_reference** | basicDataTypes:URI | 0..1 | The vocab\_reference property specifies the URI to the location of where the controlled vocabulary is defined, e.g., in an externally located XML schema file. |

## DateWithPrecisionType Class

This type is used as a replacement for the standard xs:date type but allows for the representation of the precision of the date. If the precision is given, consumers must ignore the portions of this field that is more precise than the given precision. Producers should zero-out (fill with zeros) digits in the date that are required by the xs:date datatype but are beyond the specified precision. In order to avoid ambiguity, it is strongly suggested that all dates include a specification of the timezone if it is known.

Properties of the DateWithPrecisionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **precision** | DatePrecisionEnum | 0..1 | The precision of the associated date. If omitted, the default is "day", meaning the full property value. |

## DateTimeWithPrecisionType Class

This type is used as a replacement for the standard xs:dateTime type but allows for the representation of the precision of the dateTime. If the precision is given, consumers must ignore the portions of this field that is more precise than the given precision. Producers should zero-out (fill with zeros) digits in the dateTime that are required by the xs:dateTime datatype but are beyond the specified precision. In order to avoid ambiguity, it is strongly suggested that all dateTimes include a specification of the timezone if it is known.

Properties of the DateTimeWithPrecisionType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |
| **precision** | DateTimePrecisionEnum | 0..1 | The precision of the associated dateTime. If omitted, the default is "second", meaning the full property value (including fractional seconds). |

## SIDType Class

SIDType specifies Windows Security ID (SID) types via a union of the SIDTypeEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications. Properties that use this type can express multiple values by providing them using a delimiter-separated list. The default delimiter is '##comma##' (no quotes) but can be overridden through use of the delimiter field. Note that whitespace is preserved and so, when specifying a list of values, do not include a space following the delimiter in a list unless the first character of the next list item should, in fact, be a space.

Properties of the SIDType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## Layer4ProtocolType Class

Layer4ProtocolType specifies Layer 4 protocol types, via a union of the Layer4ProtocolEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

Properties of the Layer4ProtocolType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## EndiannessType Class

The EndiannessType specifies names for byte ordering methods.

Properties of the EndiannessType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## CipherType Class

CipherType specifies encryption algorithms, via a union of the CipherEnum type and the atomic xs:string type. Its base type is the CybOX Core BaseObjectPropertyType, for permitting complex (i.e. regular-expression based) specifications.

Properties of the CipherType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## RegionalRegistryType Class

The RegionalRegistryType specifies a Regional Internet Registry (RIR) for a given WHOIS entry. RIRs defined by the RegionalRegistryTypeEnum may be used, as well as those specified by a free form text string.

Properties of the RegionalRegistryType class

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Type** | **Multiplicity** | **Description** |

## SourceClassTypeEnum Class

Table 3‑60. Literals of the SourceClassTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Network** | Describes a Network-based cyber observation. |
| **System** | Describes a System-based cyber observation. |
| **Software** | Describes a Software-based cyber observation. |

## SourceTypeEnum Class

Table 3‑61. Literals of the SourceTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Tool** | Describes a cyber observation made using various tools, such as scanners, firewalls, gateways, protection systems, and detection systems. See ToolTypeEnum for a more complete list of tools that CybOX supports. |
| **Analysis** | Describes a cyber observation made from analysis methods, such as Static and Dynamic methods. See AnalysisMethodTypeEnum for a more complete list of methods that CybOX supports. |
| **Information Source** | Describes a cyber observation made using other information sources, such as logs, Device Driver APIs, and TPM output data. See InformationSourceTypeEnum for a more complete list of information sources that CybOX supports. |

## CompensationModelEnum Class

Table 3‑62. Literals of the CompensationModelEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Freeware** | Specifies that the tool is available for use at no monetary cost as the compensation model. |
| **Shareware** | Specifies that the tool is proprietary and offers a limited use license as the compensation model. |
| **Commercial** | Specifies that the tool is produced for sale or serves commercial purposes as the compensation model. |
| **Adware** | Specifies that the tool uses automatically rendered advertisements as the compensation model. |

## ToolReferenceTypeEnum Class

Table 3‑63. Literals of the ToolReferenceTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Documentation** | The reference is to documentation about the identified tool. |
| **Source** | The reference is to source code for the identified tool. |
| **Download** | The reference is to where an executable version of the tool can be downloaded. |
| **Execute** | The reference is to the tool implemented as an online service. |
| **Other** | The reference is to material about the tool not covered by other values in this enumeration. |

## ConditionTypeEnum Class

Table 3‑64. Literals of the ConditionTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Equals** | Specifies the equality or = condition. |
| **DoesNotEqual** | Specifies the "does not equal" or != condition. |
| **Contains** | Specifies the "contains" condition. |
| **DoesNotContain** | Specifies the "does not contain" condition. |
| **StartsWith** | Specifies the "starts with" condition. |
| **EndsWith** | Specifies the "ends with" condition. |
| **GreaterThan** | Specifies the "greater than" condition. |
| **GreaterThanOrEqual** | Specifies the "greater than or equal to" condition. |
| **LessThan** | Specifies the "less than" condition. |
| **LessThanOrEqual** | Specifies the "less than or equal" condition. |
| **InclusiveBetween** | The pattern is met if the given value lies between the values indicated in the field value body, inclusive of the bounding values themselves. The field value body MUST contain at least 2 values to be valid. If the field value body contains more than 2 values, then only the greatest and least values are considered. (I.e., If the body contains "2,4,6", then an InclusiveBetween condition would be satisfied if the observed value fell between 2 and 6, inclusive. Since this is an inclusive range, an observed value of 2 or 6 would fit the pattern in this example.) As such, always treat the InclusiveBetween condition as applying to a single range for the purpose of evaluating the apply\_condition attribute. |
| **ExclusiveBetween** | The pattern is met if the given value lies between the values indicated in the field value body, exclusive of the bounding values themselves. The field value body MUST contain at least 2 values to be valid. If the field value body contains more than 2 values, then only the greatest and least values are considered. (I.e., If the body contains "2,4,6", then an InclusiveBetween condition would be satisfied if the observed value fell between 2 and 6, exclusive. Since this is an exclusive range, an observed value of 2 or 6 would not fit the pattern in this example.) As such, always treat the ExclusiveBetween condition as applying to a single range for the purpose of evaluating the apply\_condition attribute. |
| **FitsPattern** | Specifies the condition that a value fits a given pattern. |
| **BitwiseAnd** | Specifies the condition of bitwise AND. Specifically, when applying this pattern, a given value is bitwise-ANDed with the bit\_mask attribute value (which must be present). If the result is identical to the value provided in the body of this field value, the pattern is considered fulfilled. |
| **BitwiseOr** | Specifies the condition of bitwise OR. Specifically, when applying this pattern, a given value is bitwise-ORed with the bit\_mask attribute value (which must be present). If the result is identical to the value provided in the body of this field value, the pattern is considered fulfilled. |
| **BitwiseXor** | Specifies the condition of bitwise XOR. Specifically, when applying this pattern, a given value is bitwise-XORed with the bit\_mask attribute value (which must be present). If the result is identical to the value provided in the body of this field value, the pattern is considered fulfilled. |

## ConditionApplicationEnum Class

Table 3‑65. Literals of the ConditionApplicationEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **ANY** | Indicates that a pattern holds if the given condition can be successfully applied to any of the field values. |
| **ALL** | Indicates that a pattern holds only if the given condition can be successfully applied to all of the field values. |
| **NONE** | Indicates that a pattern holds only if the given condition can be successfully applied to none of the field values. |

## DatatypeEnum Class

Table 3‑66. Literals of the DatatypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **string** | Specifies the string datatype as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#string for more information. |
| **int** | Specifies the int datatype as it applies to the W3C standard for int. See http://www.w3.org/TR/xmlschema-2/#int for more information. |
| **float** | Specifies the float datatype as it apples to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#float for more information. |
| **date** | Specifies a date, which is usually in the form yyyy-mm--dd as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#date for more information. |
| **positiveInteger** | Specifies a positive integer in the infinite set {1,2,...} as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#positiveInteger for more information. |
| **unsignedInt** | Specifies an unsigned integer, which is a nonnegative integer in the set {0,1,2,...,4294967295} as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#unsignedInt for more information. |
| **dateTime** | Specifies a date in full format including both date and time as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#dateTime for more information. |
| **time** | Specifies a time as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#time for more information. |
| **boolean** | Specifies a boolean value in the set {true,false,1,0} as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#boolean for more information. |
| **name** | Specifies a name (which represents XML Names) as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#Name and http://www.w3.org/TR/2000/WD-xml-2e-20000814#dt-name for more information. |
| **long** | Specifies a long integer, which is an integer whose maximum value is 9223372036854775807 and minimum value is -9223372036854775808 as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#long for more information. |
| **unsignedLong** | Specifies an unsigned long integer, which is an integer whose maximum value is 18446744073709551615 and minimum value is 0 as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#unsignedLong for more information. |
| **duration** | Specifies a length of time in the extended format PnYn MnDTnH nMnS, where nY represents the number of years, nM the number of months, nD the number of days, 'T' is the date/time separator, nH the number of hours, nM the number of minutes and nS the number of seconds, as it applies to the W3 standard. See http://www.w3.org/TR/xmlschema-2/#duration for more information. |
| **double** | Specifies a decimal of datatype double as it is patterned after the IEEE double-precision 64-bit floating point type (IEEE 754-1985) and as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#double for more information. |
| **nonNegativeInteger** | Specifies a non-negative integer in the infinite set {0,1,2,...} as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#nonNegativeInteger for more information. |
| **hexBinary** | Specifies arbitrary hex-encoded binary data as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#hexBinary for more information. |
| **anyURI** | Specifies a Uniform Resource Identifier Reference (URI) as it applies to the W3C standard and to RFC 2396, as amended by RFC 2732. See http://www.w3.org/TR/xmlschema-2/#anyURI for more information. |
| **base64Binary** | Specifies base64-encoded arbitrary binary data as it applies to the W3C standard. See http://www.w3.org/TR/xmlschema-2/#base64Binary for more information. |
| **IPv4 Address** | Specifies an IPV4 address in dotted decimal form. CIDR notation is also accepted. |
| **IPv6 Address** | Specifies an IPV6 address, which is represented by eight groups of 16-bit hexadecimal values separated by colons (:) in the form a:b:c:d:e:f:g:h. CIDR notation is also accepted. |
| **Host Name** | Specifies a host name. For compatibility reasons, this could be any string. Even so, it is best to use the proper notation for the given host type. For example, web hostnames should be written as fully qualified hostnames in practice. |
| **MAC Address** | Specifies a MAC address, which is represented by six groups of 2 hexdecimal digits, separated by hyphens (-) or colons (:) in transmission order. |
| **Domain Name** | Specifies a domain name, which is represented by a series of labels concatenated with dots conforming to the rules in RFC 1035, RFC 1123, and RFC 2181. |
| **URI** | Specifies a Uniform Resource Identifier, which identifies a name or resource and can act as a URL or URN. |
| **TimeZone** | Specifies a timezone in UTC notation (UTC+number). |
| **Octal** | Specifies arbitrary octal (base-8) encoded data. |
| **Binary** | Specifies arbitrary binary encoded data. |
| **BinHex** | Specifies arbitrary data encoded in the Mac OS-originated BinHex format. |
| **Subnet Mask** | Specifies a subnet mask in IPv4 or IPv6 notation. |
| **UUID/GUID** | Specifies a globally/universally unique ID represented as a 32-character hexadecimal string. See ISO/IEC 11578:1996 Information technology -- Open Systems Interconnection -- Remote Procedure Call - http://www.iso.ch/cate/d2229.html. |
| **Collection** | Specifies data represented as a container of multiple data of a shared elemental type. |
| **CVE ID** | Specifies a CVE ID, expressed as CVE- appended by a four-digit integer, a - and another four-digit integer, as in CVE-2012-1234. |
| **CWE ID** | Specifies a CWE ID, expressed as CWE- appended by an integer. |
| **CAPEC ID** | Specifies a CAPEC ID, expressed as CAPEC- appended by an integer. |
| **CCE ID** | Specifies a CCE ID, expressed as CCE- appended by an integer. |
| **CPE Name** | Specifies a CPE Name. See http://cpe.mitre.org/specification/archive/version2.0/cpe-specification\_2.0.pdf for more information. |

## PatternTypeEnum Class

Table 3‑67. Literals of the PatternTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Regex** | Specifies the regular expression pattern type. |
| **Binary** | Specifies the binary (bit operations) pattern type. |
| **XPath** | Specifies the XPath 1.0 expression pattern type. |

## DataFormatEnum Class

Table 3‑68. Literals of the DataFormatEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Binary** | Specifies binary data. |
| **Hexadecimal** | Specifies hexadecimal data. |
| **Text** | Specifies text. |
| **Other** | Specifies any other type of data from the ones listed. |

## DataSizeUnitsEnum Class

Table 3‑69. Literals of the DataSizeUnitsEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Bytes** | Specifies an object size in Bytes. |
| **Kilobytes** | Specifies an object size in Kilobytes. |
| **Megabytes** | Specifies an object size in Megabytes. |

## DatePrecisionEnum Class

Table 3‑70. Literals of the DatePrecisionEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **year** | Date is precise to the given year. |
| **month** | Date is precise to the given month. |
| **day** | Date is precise to the given day. |

## TimePrecisionEnum Class

Table 3‑71. Literals of the TimePrecisionEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **hour** | Time is precise to the given hour. |
| **minute** | Time is precise to the given minute. |
| **second** | Time is precise to the given second (including fractional seconds). |

## SIDTypeEnum Class

Table 3‑72. Literals of the SIDTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **SidTypeUser** | Indicates a SID of type User. |
| **SidTypeGroup** | Indicates a SID of type Group. |
| **SidTypeDomain** | Indicates a SID of type Domain. |
| **SidTypeAlias** | Indicates a SID of type Alias. |
| **SidTypeWellKnownGroup** | Indicates a SID for a well-known group. |
| **SidTypeDeletedAccount** | Indicates a SID for a deleted account. |
| **SidTypeInvalid** | Indicates an invalid SID. |
| **SidTypeUnknown** | Indicates a SID of unknown type. |
| **SidTypeComputer** | Indicates a SID for a computer. |
| **SidTypeLabel** | Indicates a mandatory integrity label SID. |

## Layer4ProtocolEnum Class

Table 3‑73. Literals of the Layer4ProtocolEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **TCP** | Specifies the Transmission Control Protocol. |
| **UDP** | Specifies the User Datagram Protocol. |
| **AH** | Specifies the Authentication Header protocol. |
| **ESP** | Specifies the Encapsulating Security Payload protocol. |
| **GRE** | Specifies the Generic Routing Encapsulation protocol. |
| **IL** | Specifies the Internet Link protocol. |
| **SCTP** | Specifies the Stream Control Transmission Protocol. |
| **Sinec H1** | Specifies the Siemens Sinec H1 protocol. |
| **SPX** | Specifies the Sequenced Packet Exchange protocol. |
| **DCCP** | Specifies the Datagram Congestion Control Protocol. |

## EndiannessTypeEnum Class

Table 3‑74. Literals of the EndiannessTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **Big-endian** | The Big-endian value specifies a big-endian byte ordering. |
| **Little-endian** | The Little-endian value specifies a little-endian byte ordering. |
| **Middle-endian** | The Middle-endian value specifies a middle-endian byte ordering. |

## CipherEnum Class

Table 3‑75. Literals of the CipherEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **3DES** | Specifies the Triple Data Encryption Standard (DES) algorithm. |
| **AES** | Specifies the Advanced Encryption Standard (AES) algorithm. |
| **Blowfish** | Specifies the Blowfish algorithm. |
| **CAST-128** | Specifies the CAST-128 algorithm. |
| **CAST-256** | Specifies the CAST-256 algorithm. |
| **DES** | Specifies the Data Encryption Standard (DES) algorithm. |
| **IDEA** | Specifies the International Data Encryption Algorithm (IDEA). |
| **Rijndael** | Specifies the Rijndael algorithm. |
| **RC5** | Specifies the RC5 algorithm. |
| **Skipjack** | Specifies the Skipjack algorithm. |

## RegionalRegistryTypeEnum Class

Table 3‑76. Literals of the RegionalRegistryTypeEnum enumeration

|  |  |
| --- | --- |
| **Enumeration Literal** | **Description** |
| **AfriNIC** | AfriNIC stands for African Network Information Centre, and is the RIR for Africa. |
| **ARIN** | ARIN stands for American Registry for Internet Numbers, and is the RIR for the United States, Canada, several parts of the Caribbean Region, and Antarctica. |
| **APNIC** | APNIC stands for Asia-Pacific Network Information Centre, and is the RIR for Asia, Australia, New Zealand, and neighboring countries. |
| **LACNIC** | LACNIC stands for Latin American and Caribbean Network Information Centre, and is the RIR for Latin America and parts of the Caribbean region. |
| **RIPE NCC** | RIPE NCC stands for Réseaux IP Européens Network Coordination Centre, and is the RIR for Europe, Russia, the Middle East, and Central Asia. |