|  |  |
| --- | --- |
| January 24, 2011 |  |
|  | Security Framework |
| Version 1.0 |  |
| Security Monitor user's guide | |



Table of Contents

1 Introduction 1

2 Configuration files structure 1

2.1 Security Monitor configuration reference 1

2.1.1 Attributes 1

2.1.2 Remarks 2

2.2 Security Framework common configuration file structure 2

2.3 Security Monitor configuration file structure 3

**2.3.1** SecurityMonitorProperties 4

2.3.2 Recorders 5

2.3.3 Filters 8

2.3.4 Policies 9

2.3.5 Rules 10

2.3.6 Actions 16

2.3.7 LogAnalyzer 23

3 Security Monitor API 24

3.1 API Methods 24

3.2 API Examples 26

3.2.1 Defining custom filter for an event recorder 26

3.2.2 Defining Action handler 26

3.2.3 Registering action handler 27

3.2.4 Configuring Security Monitor policy 28

3.2.5 Checking data for SQL injection 28

3.2.6 Reporting statistics events 29

# Introduction

Security Monitor provides a toolset for reporting, collecting and processing security related events. It analyzes the events collections and allows detecting suspicious and malicious activities those can’t be detected by individual operations analyses. Then the Security Monitor makes a decision which action(s) should be performed in response on the malicious activity if found.

Security Monitor provides API that allows reporting events, registering action handlers, modifying configuration on fly.

The analysis is based on configurable policies. Each policy consists of a set of rules and a set of actions to be performed if a security event matches the policy.

The Security Monitor passes the information on recommended responding actions via event handlers those are registered for especial action type(s) by the consumer.

Actually the configurations can be specified only in the configuration files.

# Configuration files structure

Configuration is a set of XML documents. Each Security Framework subsystem has a separate configuration file. There is a root configuration file - MtSfConfigurationLoader.xml that references all subsystem’s configuration files.

## Security Monitor configuration reference

The root configuration file references the Security Monitor configuration and Common Security Framework configuration (which is important for Security Monitor) as it’s shown below:

<ConfigurationLoader>

<Items>

<item Name="SecurityFrameworkSettings">

|  |
| --- |
| <item Id="766DCC7A-DDBB-4759-A74A-1BE9611EA774" IsServiceSubsystem="true" IsEnabled="true" Name="SecurityFrameworkSettings" Path="MtSfSettings.xml" Type="MetraTech.SecurityFramework.Core.Common.Configuration.SecurityFrameworkSettings" /> |

<item Name="SecurityMonitor">

|  |
| --- |
| <item Id="5A3640F5-EFB1-4437-B450-633DEDCD60CD" IsEnabled="true" Name="SecurityMonitor" Path="MtSfSsSecurityMonitor.xml" SubsystemType="MetraTech.SecurityFramework.Core.SecurityMonitor.SecurityMonitor" Type="MetraTech.SecurityFramework.Core.SecurityMonitor.SecurityMonitorProperties"/> |

### Attributes

| Attribute | Description |
| --- | --- |
| Id | Required attribute.  Defines a unique item ID. Uses internally. It must be GUID. |
| IsServiceSubsystem | Optional attribute.  Indicates that a subsystem is important for other subsystems. Uses internally. |
| IsEnabled | Required attribute.  Indicates whether the subsystem is enabled. Uses for turning the subsystem off when it is not needed. |
| Name | Required attribute.  Defines the subsystem name. Uses internally. |
| Path | Required attribute.  Defines a path to the subsystem configuration file. This path is relative to the root configuration file. I.e., when the subsystem configuration file is placed to the same folder the root configuration file is, simply provide a file name only. |
| SubsystemType | Required attribute for the Security Monitor configuration reference.  Defines a type that provides an access to the subsystem functionality. This value has not to be changed. |
| Type | Required attribute.  Defines a type that takes the subsystem properties when it is loaded from the configuration file. This value has not to be changed. |

### Remarks

The really important attributes are the Path and IsEnabled. First of them indicates a file with the configuration and second allows turning the subsystem off. You may want to turn the Security Monitor off if it causes performance problems, for example.

## Security Framework common configuration file structure

The common configuration file structure is shown below:

|  |
| --- |
| <SecurityFrameworkSettings  autoMonitorExceptions="true"  executionContextTypeName="MetraTech.SecurityFramework.Core.Common.Web.HttpExecutionContext, MetraTech.SecurityFramework">  </SecurityFrameworkSettings> |

Now it contains only root element – SecurityFrameworkSettings. It’s not required but recommended do not change the element’s name. It matches with name of the class that takes the configuration properties when it is loading.

The root element has the following attributes

| Attribute | Description |
| --- | --- |
| AutoMonitorExceptions | Optional attribute.  Indicates whether security issues found by any of the Security Framework’s engines are reported to the Security Monitor automatically and the consumer need not to make such reporting. |
| ExecutionContextTypeName | Required attribute.  Defines a type of class that implements MetraTech.SecurityFramework.Core.Common.IExecutionContext and provides an access to some context specific things such as Client IP address, Client Session ID and so on.  Different kinds of apps require different IExecutionContext implementations. For example, WEB app requires one implementation and Windows desktop app requires another.  Actually the Security Framework provides two implementations:   1. MetraTech.SecurityFramework.Core.Common.Web.HttpExecutionContext – for WEB apps 2. MetraTech.SecurityFramework.Core.Common.Testing.UnitTestExecutionContext – for testing purposes. |

## Security Monitor configuration file structure

The Security Monitor configuration file structure at high level is shown below:

|  |
| --- |
| <SecurityMonitorProperties IsRuntimeApiEnabled="True|False" IsRuntimeApiPublic="True|False" IsControlApiEnabled="True|False" IsControlApiPublic="True|False" RecordInputData="True|False" RecordEventReason="True|False">  <Recorders>  <item >  <Filters>  </Filters>  </item>  <item >  <Filters>  </Filters>  </item>  </Recorders>  <Policies>  <item >  <Rules>  <item />  <item />  </Rules>  <Actions>  <item />  <item />  </Actions>  </item>  </Policies>  <LogAnalyzer />  </SecurityMonitorProperties> |

The following sections describe attributes, elements, and child elements.

### **SecurityMonitorProperties**

Root element. It’s not required but recommended do not change the element’s name. It matches with name of the class that takes the configuration properties when it is loading.

There are the following child elements and attributes under the root.

Attributes:

| Attribute | Description |
| --- | --- |
| IsRuntimeApiEnabled | Required attribute.  Indicates whether the Security Monitor API is enabled. Prevents the API from initializing if set to **false**. |
| IsRuntimeApiPublic | Required attribute.  Indicates whether the Security Monitor API is exposed to public access. |
| IsControlApiEnabled | Required attribute.  Indicates whether the Security Monitor control API is enabled. Prevents the control API from initializing if set to **false**. *For future use*. |
| IsControlApiPublic | Required attribute.  Indicates whether the Security Monitor control API is exposed to public access. *For future use*. |
| RecordInputData | Optional attribute.  Indicates whether an input data should be recorded into the Security Monitor logs. Allows preventing sensitive data from falling to the public place. |
| RecordEventReason | Optional attribute.  Indicates whether a security event reason should be recorded into the Security Monitor logs. Allows preventing sensitive data from falling to the public place. |

Elements:

| Element | Description |
| --- | --- |
| Recorders | Required element.  Defines recorders (writers) those collect security events for subsequent analyses. |
| Policies | Required element.  Defines rule sets those allow making a decision which actions should be performed in response on each of security events. |
| LogAnalyzer | Required element.  Defines a layer to access the security events log to be used by security policy rules. It performs any calculations based on previously logged security events. |

The enlisted elements are described below.

### **Recorders**

Defines recorders (writers) those collect security events for subsequent analyses. The element represents a collection of recorder definitions (<item> elements). Each child element defines a recorder and has attributes specific to that recorder.

The following recorders are already defined in the Security Monitor subsystem:

| Recorder | Description |
| --- | --- |
| CSV file recorder | Writes security events into a CSV file. Allows to choice which columns will are written. Allows rolling the output file on daily basis. |
| SQLite database recorder | Writer security events into a SQLite database for subsequent analyses. |

Each recorder has its own attributes those have to be configured, which are described below:

**CSV file recorder**

<SecurityMonitorProperties>

<Recorders>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.CsvFileRecorderDefinitionProperties, MetraTech.SecurityFramework"  Id="<unique\_ID>"  LogFileName="<path\_to\_file>"  RollFileName="True|False">  <Filters>  <item />  </Filters>  <Columns>  <item Name="EventType|SubsystemName|CategoryName|TimeStamp|Path|HostName|Message|ClientAddress|UserIdentity|SessionId|ClientInfo|StackTrace|InputData|Reason" Caption="<Friendly name>" />  </Columns>  </item> |

Child elements and attributes for CSV file recorder are described below.

Elements:

| Element | Description |
| --- | --- |
| Filters | Optional element.  Defines filters those allow to limit security events logged by some conditions. A security event must match at least one of defined filters to be passed to recorder.  When no filters are defined all security event are recorded.  On the other hand, if only one filter is defined events are recorded when they match this filter only. |
| Columns | Required element.  Defines friendly names for columns in the CSV file and allows excluding some columns from the file. Only columns defined under the **Columns** element will appear in the file. Also it defines a columns order in the file.  Warning! If you have changed the Columns settings, it’s recommended to rename or delete current CSV file, because it become inconsistent with new settings. |

Attributes:

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a recorder type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.CsvFileRecorderDefinitionProperties, MetraTech.SecurityFramework” |
| Id | Required attribute.  A unique ID of the recorder. The recorder definition can be gotten by this value via the Security Monitor API. |
| LogFileName | Required attribute.  Defines a full path to the log file.  **The path should be absolute. The directory must exist and the account the application is ran under must have Modify permissions for this directory.** |
| RollFileName | Optional attribute.  Indicates that the recorder creates rolls the output file on daily basis, i.e. creates one file per day. |

**SQLite database recorder**

Child elements and attributes for SQLite database recorder are described below.

Elements:

| Element | Description |
| --- | --- |
| Filters | Optional element.  Defines filters those allow to limit security events logged by some conditions. A security event must match at least one of defined filters to be passed to recorder. |

Attributes:

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a recorder type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.SQLiteRecorderDefinitionProperties, MetraTech.SecurityFramework” |
| Id | Required attribute.  A unique ID of the recorder. The recorder definition can be gotten by this value via the Security Monitor API. |
| DatabaseFileName | Required attribute.  Defines an app relative or absolute path to the database file. For WEB apps it should be started with ~ character. For example, ~/App\_Data/SecurityFramework.db or c:\data\ SecurityFramework.db |
| AsynchronousRecording | Optional attribute.  Indicates to use asynchronous recording into SQLite database.  **Important NOTE!** Setting this attribute to True significantly increases the performance but may cause the database damage if Windows or power source falls down.  Value: Boolean (default False) |

### **Filters**

Represents a collection of filters those can be to security events to limit the log size.

<SecurityMonitorProperties>

<Recorders>

<item>

<Filters>

|  |
| --- |
| <item EventType="Unknown|WebRequestEventType|WebResponseEventType|AuthenticationEventType|AccessControlEventType|SessionEventType|InputDataProcessingEventType|OutputDataProcessingEventType|FileIoEventType|UserActivityTrendEventType|AppActivityTrendEventType"  SubsystemName="<Legal\_subsystem\_name>" SubsystemCategoryName="Subsystem\_category\_name">  <CustomFilter TypeName="<Hanlder\_type\_name>" MethodName="<Handler\_method>" />  </item> |

Child elements and attributes for individual filter item are described below.

Elements:

| Element | Description |
| --- | --- |
| CustomFilter | Optional element.  Defines a custom filter event handler. The element point to a static method of some class within the app. The method must fit the CustomFilterEventHandler delegate type. It has 2 required attributes: TypeName – defines an assembly or fully qualified type name of the class containing the handler method, MethodName – defines a public static method name that provides the filtering. |

Attributes:

| Attribute | Description |
| --- | --- |
| EventType | Optional attribute.  Defines a security event type to be used for the filter. It can be set to one of following values: Unknown, WebRequestEventType, WebResponseEventType, AuthenticationEventType, AccessControlEventType, SessionEventType, InputDataProcessingEventType, OutputDataProcessingEventType, FileIoEventType, UserActivityTrendEventType, AppActivityTrendEventType.  Unknown indicates that any of types matches the filter. |
| SubsystemName | Optional attribute.  Defines a source subsystem name to be used for the filter. It can be any legal (existing) subsystem name. |
| SubsystemCategoryName | Optional attribute.  Defines a source subsystem’s category to be used for the filter. It can be any category from the subsystem defined in the **SubsystemName** attribute. When this value is omitted, the filter does not limit security events by their source category. |

### Policies

Defines a collection of the Security Monitor policies those allow make a decision which action should be performed in response on a security event.

<SecurityMonitorProperties>

<Policies>

|  |
| --- |
| <item Id="<unique\_ID>">  <Rules>  <item />  <item />  </Rules>  <Actions>  <item />  <item />  </Actions>  </item> |

Each child element defines a single policy and has the following child elements and attributes.

Elements:

| Element | Description |
| --- | --- |
| Rules | Required element.  Defines a collection of security event processing rules to be evaluated for each reported security event. There are several rule types those are described below. |
| Actions | Required element.  Defines a collection of actions to be performed in response on the processed security event that matches all policy rules.  **When several actions of the same type are generated during security event processing, only one of them is passed to a calling app.** |

Attributes:

| Attribute | Description |
| --- | --- |
| Id | Required element.  A unique ID of the policy. Uses internally. |

Two next sections describe Policy configuration elements.

### Rules

Defines a collection of security event processing rules. Any processing security event must match all rules in the policy’s Rules collection to match the policy and a decision to perform specified actions is made.

The following security policy rule types are already defined in the Security Monitor subsystem:

| Rule | Description |
| --- | --- |
| Event Type Rule | Provides filtering of events by their Event Type. |
| Event Source Rule | Provides filtering of events by their source subsystem and subsystem’s category. |
| **Event Repeat Threshold Rule** | Provides filtering by a number of events coming from the same source (subsystem and category) within the same user session. |
| ****Event Time span Threshold Rule**** | Provides filtering by a number of events coming from the same source (subsystem and category) within the specified time frame. |
| ****IP Change Rule**** | Provides filtering of events by a number of IP addresses used in a single session. |
| ****Action Frequency Rule**** | Provides filtering of events by a time elapsed since some policy action was recommended. |
| ****Input Data Size Rule**** | Provides filtering by an input data size. |
| ****Event Ration Rule**** | Provides filtering by a ratio of events coming from one source to events from another source in a specified period of time. |

**Event Type Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventTypeRule, MetraTech.SecurityFramework"  EventType="Unknown|WebRequestEventType|WebResponseEventType|AuthenticationEventType|AccessControlEventType|SessionEventType|InputDataProcessingEventType|OutputDataProcessingEventType|FileIoEventType|UserActivityTrendEventType|AppActivityTrendEventType"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventTypeRule, MetraTech.SecurityFramework” |
| EventType | Required attribute.  Defines event type (category) to be matched this rule. It can any of the following values: Unknown, WebRequestEventType, WebResponseEventType, AuthenticationEventType, AccessControlEventType, SessionEventType, InputDataProcessingEventType, OutputDataProcessingEventType, FileIoEventType, UserActivityTrendEventType, AppActivityTrendEventType |

**Event Source Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventSourceRule, MetraTech.SecurityFramework"  SubsystemName="<Source\_subsystem\_name>"  CategoryName="<Source\_category\_name>"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventSourceRule, MetraTech.SecurityFramework” |
| SubsystemName | Required attribute.  Defines event source subsystem name to be matched this rule. It can be any of legal subsystem names. The comparison is case insensitive. |
| CategoryName | Optional attribute.  Defines event source subsystem category name to be matched this rule. It can be any of categories from within the subsystem defined in **SubsystemName** attribute. The comparison is case insensitive. If this value is omitted only category name will be used to define whether event matches the rule. |

**Event Repeat Threshold Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventRepeatThresholdRule, MetraTech.SecurityFramework"  RepeatThreshold="<Integer\_number>"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy. EventRepeatThresholdRule, MetraTech.SecurityFramework” |
| RepeatThreshold | Required attribute.  Defines a number of occurrences of events coming from the same source (subsystem and category) that is allowed to happen within the same user session until the rule is matched. I.e., any next reporting event with the same source causes rule matching. |

**Event Time span Threshold Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventTimespanThresholdRule, MetraTech.SecurityFramework"  RepeatThreshold="<Integer\_number>"  TimeThreshold="HH:mm:ss"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy. EventTimespanThresholdRule, MetraTech.SecurityFramework” |
| RepeatThreshold | Required attribute.  Defines a number of occurrences of events coming from the same source (subsystem and category) that is allowed to happen within the specified time window until the rule is matched. I.e., any next reporting event with the same source causes rule matching. |
| TimeThreshold | Required attribute.  Defines a time window to count events within. |

**IP Change Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.IPChangeRule"  NumberThreshold="<Integer\_Number>"></item> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.IPChangeRule” |
| NumberThreshold | Optional attribute.  Defines a number of IP addresses can be used within a single session until rule matches.  Value: positive integer number (default 1). |

**Action Frequency Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.ActionFrequencyRule"  PolicyActionType="BlockOperation|BlockUser|BlockAddress|Log|RedirectOperation|RedirectUser|LogoutUser|SendSecurityWarningToUser|SendAdminNotification|ChangeSessionParameter"  TimeThreshold="HH:mm:ss"></item> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.ActionFrequencyRule” |
| PolicyActionType | Required attribute.  Defines an action type to count a period of time since last invocation.  Value can be any of the following:  BlockOperation, BlockUser, BlockAddress, Log, RedirectOperation, RedirectUser, LogoutUser, SendSecurityWarningToUser, SendAdminNotification, ChangeSessionParameter |
| TimeThreshold | Required attribute.  Defines a minimum delay between invocations of actions of the specified type. |

**Input Data Size Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.InputDataSizeRule"  SizeLimit="<Integer\_number>"></item> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.InputDataSizeRule” |
| SizeLimit | Required attribute.  Defines a max size of input data.  Value: positive integer number. |

**Event Ratio Rule**

<SecurityMonitorProperties>

<Policies>

<item>

<Rules>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventRatioThresholdRule"  TimeThreshold="HH:mm:ss"  RatioLimit="<Floating\_point\_number>"  SourceSubsystem1="<Source\_1\_subsystem\_name>"  SourceCategory1="<Source\_1\_category\_name>"  SourceSubsystem2="<Source\_2\_subsystem\_name>"  SourceCategory2="<Source\_2\_category\_name>"></item> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a rule type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventRatioThresholdRule” |
| TimeThreshold | Required attribute.  Defines a minimum delay between invocations of actions of the specified type. |
| RatioLimit | Required attribute.  Defines a ratio of a number of events coming from one source to a number of events coming from another source threshold.  Value: floating point number. |
| SourceSubsystem1 | Required attribute.  Defines a subsystem part of first source.  Value: any valid subsystem name. |
| SourceCategory1 | Required attribute.  Defines a category part of first source.  Value: any valid category for the subsystem specified by the “SourceSubsystem1” attribute. |
| SourceSubsystem2 | Required attribute.  Defines a subsystem part of second source.  Value: any valid subsystem name. |
| SourceCategory2 | Required attribute.  Defines a category part of second source.  Value: any valid category for the subsystem specified by the “SourceSubsystem2” attribute. |

### Actions

Defines a collection of actions to be recommended to performing by a calling app in response to the processing security event.

The following security policy action types are already defined in the Security Monitor subsystem:

| Action | Description |
| --- | --- |
| Block Operation | Indicates that the operation causing the security event should be blocked for a specified period. |
| Block User | Indicates that the user causing the security event should be blocked for a specified period. |
| Block Address | Indicates that the IP address causing the security event should be blocked for a specified period. |
| Log | Indicates that the security event should be recorded into the app log. |
| Redirect Operation | Indicates that the user’s request should be redirected to the specified location. |
| Redirect User | Indicates that all operations of the user causing the security event should be redirected to the specified location. |
| Logout User | Indicates that the user causing the security event should be logged out. |
| Send Security Warning to User | Indicates that the specified message should be displayed to the user causing the security event. |
| Send Notification to Admin | Indicates that the specified message should be sent to an app’s administration. |
| Change Session Parameter | Indicates that the specified parameter should be changed for the user’s session causing the security event. |

Each action type has its own attributes those have to be configured, which are described below:

**Block Operation**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.BlockOperationPolicyAction, MetraTech.SecurityFramework"  BlockingPeriod="HH:mm:ss"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.BlockOperationPolicyAction, MetraTech.SecurityFramework” |
| BlockingPeriod | Required attribute.  Defines the period to block the operation for. The value must be in the format HH:mm:ss |

**Block User**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy. BlockUserPolicyAction, MetraTech.SecurityFramework"  BlockingPeriod="HH:mm:ss"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.BlockUserPolicyAction, MetraTech.SecurityFramework |
| BlockingPeriod | Required attribute.  Defines the period to block the user for. The value must be in the format HH:mm:ss |

**Block Address**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.BlockAddressPolicyAction, MetraTech.SecurityFramework"  BlockingPeriod="HH:mm:ss"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.BlockAddressPolicyAction, MetraTech.SecurityFramework” |
| BlockingPeriod | Required attribute.  Defines the period to block the IP address for. The value must be in the format HH:mm:ss |

**Log**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy. LogPolicyAction, MetraTech.SecurityFramework"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.LogPolicyAction, MetraTech.SecurityFramework” |

**Redirect Operation**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.RedirectOperationPolicyAction" DestinationPath="<Desired\_location>"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.RedirectOperationPolicyAction, MetraTech.SecurityFramework” |
| DestinationPath | Required attribute.  Defines the location to redirect the operation causing the security event. |

**Redirect User**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.RedirectUserActionPolicy" DestinationPath="<Desired\_location>"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.RedirectUserActionPolicy, MetraTech.SecurityFramework” |
| DestinationPath | Required attribute.  Defines the location to redirect all requests from the user causing the security event. |

**Logout User**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy. LogoutUserPolicyAction" /> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.LogoutUserPolicyAction, MetraTech.SecurityFramework” |

**Send Security Warning to User**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.NotifyUserPolicyAction"  Message="<Message\_text>"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.NotifyUserPolicyAction, MetraTech.SecurityFramework” |
| Message | Required attribute.  Defines a message to be displayed to the user causing the security event. |

**Send Notification to Admin**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.NotifyAdminPolicyAction"  Message="Test message"  AdminEmailAddress="admin@domain.com"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.NotifyUserPolicyAction, MetraTech.SecurityFramework” |
| Message | Required attribute.  Defines a message to be send to the app administrator. |
| AdminEmailAddress | Required attribute.  Defines an email address of the administration to be notified. |

**Change Session Parameter**

<SecurityMonitorProperties>

<Policies>

<item>

<Actions>

|  |
| --- |
| <item RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.ChangeSessionParameterPolicyAction"  ParameterName="Param"  ParameterValue="Value"/> |

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines an action type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.ChangeSessionParameterPolicyAction, MetraTech.SecurityFramework” |
| ParameterName | Required attribute.  Defines session parameter name to be changed. |
| ParameterValue | Required attribute.  Defines a value to be set to the specified session parameter. |

### LogAnalyzer

Defines a security events log reader. The reader is used by security policy rules (such as **Event Repeat Threshold Rule** and **Event Time span Threshold Rule**) to evaluate their conditions.

There is only one reader is already defined in the Security Monitor subsystem:

<SecurityMonitorProperties>

|  |
| --- |
| <LogAnalyzer  RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.SQLiteLogAnalyzer, MetraTech.SecurityFramework"  DatabaseFileName="<App\_relative\_DB\_path>"/> |

Attributes for SQLite event log analyzer are described below:

| Attribute | Description |
| --- | --- |
| RealType | Required attribute.  Defines a recorder type. Must be “MetraTech.SecurityFramework.Core.SecurityMonitor.SQLiteLogAnalyzer, MetraTech.SecurityFramework” |
| DatabaseFileName | Required attribute.  Defines an app relative or absolute path to the database file. For WEB apps it should be started with ~ character. For example, ~/App\_Data/SecurityFramework.db or c:\data\ SecurityFramework.db |

# Security Monitor API

Security Monitor subsystem can be accessed via the SecurityMonitor property of the MetraTech.SecurityFramework.SecurityKernel class.

The subsystem class has two properties Api and ControlApi. So call MetraTech.SecurityFramework.SecurityKernel.SecurityMonitor.Api to access the Security Monitor API.

## API Methods

The API provides the following methods:

| Method | Description |
| --- | --- |
| void ReportEvent(ISecurityEvent evt) | Processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void AddPolicyActionHandler(string idHandler, SecurityPolicyActionType actionType, ISecurityPolicyActionHandler handler) | Adds an action handler for a specific action type(s).  Arguments:  idHandler - An ID of the handler. Uses to remove the handler later.  actionType - Action type(s) to add the handler for.  handler – The action handler. It must be an instance of class that implements ISecurityPolicyActionHandler interface. |
| void RemovePolicyActionHandler(string idHandler) | Removes an action handler with a specified ID.  Arguments:  idHandler - An ID of the handler to be remover. A handler with such ID must be registered. |
| EventRecorderDefinition GetRecorder(string idRecorder) | Gets an event recorder with a specified ID. Uses to set custom event filter(s) if it’s necessary.  Arguments:  idRecorder - An ID of the recorder to be retrieved.  Returns a recorder definition if found and throws an exception otherwise. |
| void ReportWebRequestEvent(ISecurityEvent evt) | Sets event type to WebRequestEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportAuthenticationEvent(ISecurityEvent evt) | Sets event type to AuthenticationEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportAccessControlEvent(ISecurityEvent evt) | Sets event type to AccessControlEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportSessionEvent(ISecurityEvent evt) | Sets event type to SessionEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportInputDataProcessingEvent(ISecurityEvent evt) | Sets event type to InputDataProcessingEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportOutputDataProcessingEvent(ISecurityEvent evt) | Sets event type to OutputDataProcessingEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportFileIoEvent(ISecurityEvent evt) | Sets event type to FileIoEventType and processes an event and records it to the storage.  Arguments:  evt – The security event to be processed. |
| void ReportLogin() | Processes a successful user login security event. Must be called after the user authentication. |
| void ReportLogout() | Processes an explicit user logout security event. Must be called before user is logged out. |
| void ReportFeatureUsage(string featurePath) | Processes feature usage security events.  Agruments:  featurePath – A path to the feature that was used. |
| void ReportTransactionUsage(string featurePath) | Processes a transaction usage security event.  Agruments:  featurePath – A path to the feature that invoked a transaction. |
| void ReportIrregularUsage(string featurePath) | Processes an irregular feature usage security event.  Agruments:  featurePath – A path to the feature the was used in unexpected way. |
| void ReportFileUpload(ulong fileSize) | Processes a file upload security event.  Agruments:  fileSize – A size of the uploaded file. |

## API Examples

Next sections show how to use Security Monitor API.

WEB page that displays a security warning to the user is considered as an example. Also, some custom filter added to the security event recorder definition.

### Defining custom filter for an event recorder

Event custom filter is implemented as a static method of a static class.

It discards all events coming from Encoder subsystem during recording.

internal static class TestCustomFilter

{

public static void Filter(object sender, CustomFilterEventArgs e)

{

e.Matched = e.SecurityEvent.SubsystemName == "Encoder";

}

}

### Defining Action handler

You need implement ISecurityPolicyActionHandler interface. It is done directly in the page class.

public partial class \_Default : System.Web.UI.Page, ISecurityPolicyActionHandler

{

public void Handle(PolicyAction policyAction, ISecurityEvent securityEvent)

{

NotifyUserPolicyAction action = policyAction as NotifyUserPolicyAction;

if (action != null)

{

custPolicy.ErrorMessage = action.Message;

custPolicy.IsValid = false;

}

RedirectOperationPolicyAction redirectAction = policyAction as RedirectOperationPolicyAction;

if (redirectAction != null)

{

Response.Redirect(redirectAction.DestinationPath);

}

}

…

}

This action handler can handle actions of two types: **Send Security Warning to User** and **Redirect Operation**. Where custPolicy – is a **CustomValidator** object.

### Registering action handler

After the action handler is created it’s necessary to register it. An appropriate place to register a handler is a OnInit page event handler.

protected void Page\_Init(object sender, EventArgs e)

{

if (SecurityKernel.IsSecurityMonitorEnabled && SecurityKernel.SecurityMonitor.IsRuntimeApiEnabled)

{

SecurityKernel.SecurityMonitor.Api.AddPolicyActionHandler(

"MyApp",

SecurityPolicyActionType.SendSecurityWarningToUser | SecurityPolicyActionType.RedirectOperation,

this);

}

}

The action handler is registered for 2 types of actions in this example.

Because of the WEB page is a security policy action handler itself and a page instance is destroyed immediately after the user’s request is processed you need to unregister a handler before the page is destroyed.

protected void Page\_Unload(object sender, EventArgs e)

{

if (SecurityKernel.IsSecurityMonitorEnabled && SecurityKernel.SecurityMonitor.IsRuntimeApiEnabled)

{

SecurityKernel.SecurityMonitor.Api.RemovePolicyActionHandler("MyApp");

}

}

### Configuring Security Monitor policy

Here is an example of a simple security monitor policy that generates a **Send Security Warning to User** action in response on a SQL injection.

<SecurityMonitorProperties Signature="">

<SubsystemProps IsRuntimeApiEnabled="true" IsRuntimeApiPublic="true" IsControlApiEnabled="true" IsControlApiPublic="true" />

<Recorders>

<item

RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.SQLiteRecorderDefinitionProperties, MetraTech.SecurityFramework"

Id="SQLiteRecorder"

DatabaseFileName="..\..\..\Tests\SecurityFrameworkUnitTests\App\_Data\SecurityFramework.db">

<Filters>

</Filters>

</item>

</Recorders>

<Policies>

<item Id="SqlDetectionPolicy">

<Actions>

<item

RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.NotifyUserPolicyAction, MetraTech.SecurityFramework"

Message="SQL code is not allowed here!"></item>

</Actions>

<Rules>

<item

RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.Policy.EventSourceRule, MetraTech.SecurityFramework"

SubsystemName="Detector"

CategoryName="Sql"></item>

</Rules>

</item>

</Policies>

<LogAnalyzer

RealType="MetraTech.SecurityFramework.Core.SecurityMonitor.SQLiteLogAnalyzer, MetraTech.SecurityFramework"

DatabaseFileName="..\..\..\Tests\SecurityFrameworkUnitTests\App\_Data\SecurityFramework.db"/>

</SecurityMonitorProperties>

### Checking data for SQL injection

A simplest way to check an input data for SQL injection is call DetectSql extension method.

try

{

string inputText = TextBoxInputData.Text;

if (!string.IsNullOrEmpty(inputText))

{

inputText.DetectSql();

}

}

catch (DetectorInputDataException x)

{

// Some XSS injection found

…

Where TextBoxInputData - is a **TextBox** object.

### Reporting statistics events

using MetraTech.SecurityFramework;

…

SecurityKernel.SecurityMonitor.Api.ReportLogin();

…

SecurityKernel.SecurityMonitor.Api.ReportLogout();

…

SecurityKernel.SecurityMonitor.Api.ReportFileUpload(1000000);