About This Guide

Using the EventTracker API, third parties can integrate the EventTracker solution into their own applications using an extensible XML interface. This user guide is intended for application developers who will use the EventTracker API.

Scope

This is a technical document and is intended for use by developers with experience in various aspects of log, change and configuration management. Only information specific to the use of EventTracker API is provided.

Audience

This user guide is intended for application developers who will use the EventTracker API v7.x.
# Table of Contents

Preface ................................................................................................................................. 6
About Prism Microsystems, Inc. ................................................................................................. 6
Contact Prism Support .............................................................................................................. 7
EventTracker API Overview ..................................................................................................... 7
Services Architecture ................................................................................................................ 8
  Step 1 - Receives an HTTPS Request .................................................................................. 8
  Step 2 - Performs an EventTracker Function ...................................................................... 8
  Step 3 - Returns an XML Report .......................................................................................... 8
EventTracker User Account ...................................................................................................... 9
Decoding XML Reports ............................................................................................................ 10
Authentication .......................................................................................................................... 10
Deploy EventTracker API ....................................................................................................... 11
  Summary ............................................................................................................................. 11
  Set up Virtual Directory- Windows server 2003 (V6) .......................................................... 12
    Configure Virtual Directory Properties ........................................................................... 17
    Browse the Virtual Directory ....................................................................................... 21
  Set up Virtual Directory- Windows Server 2008 (Version-7.0, 7.5) .................................... 22
EventTracker Conventions ....................................................................................................... 27
  Supported Operations ......................................................................................................... 27
  Xml Structure ....................................................................................................................... 27
  Xml Nodes/Attributes ......................................................................................................... 27
  Web Service Files ................................................................................................................ 27
EventTracker Web Services ..................................................................................................... 28
  EventTracker.Configs.Alert ............................................................................................... 28
    GetAlert ............................................................................................................................ 28
    GetAlerts .......................................................................................................................... 28
  EventTracker.Configs.Category ........................................................................................... 28
    GetAllCategories ........................................................................................................... 28
    GetCategory .................................................................................................................... 29
    GetCategoryByGroup ...................................................................................................... 29
    GetCategoryHierarchy .................................................................................................... 29
    GetCompleteHierarchy .................................................................................................. 30
EventTracker.Configs.CollectionPoint ................................................................. 30
    GetCollectionPointCabStatus ........................................................................... 30
    GetCollectionPointDetail .................................................................................. 31
    GetCollectionPointManagers ............................................................................ 31
EventTracker.Configs.EAM .................................................................................. 31
    GetConfigs ........................................................................................................... 31
    GetCustomColumns ............................................................................................. 32
    GetCustomRule ..................................................................................................... 32
    GetCustomRules .................................................................................................... 33
    GetCustomRulesByStatus .................................................................................... 34
    GetCustomRulesColumns ..................................................................................... 34
    GetCustomRulesEvents ....................................................................................... 34
EventTracker.Configs.EventVault .......................................................................... 34
    GetCabInfo .......................................................................................................... 35
    GetCabInfoSystem ............................................................................................... 35
EventTracker.Configs.KeyValuePair .................................................................... 35
    GetKeyValuePair .................................................................................................... 35
EventTracker.Configs.Manager ............................................................................. 37
    GetDLAConfiguration .......................................................................................... 37
    GetEmail .............................................................................................................. 37
    GetNetFlowReceiver .......................................................................................... 37
    GetSyslogPorts .................................................................................................... 38
    GetVCPPorts ....................................................................................................... 38
EventTracker.Configs.ReceiverFilter ................................................................... 38
    GetReceiverFilter ............................................................................................... 38
    GetReceiverFilters ............................................................................................. 38
    GetCostSavingAnalysis ....................................................................................... 39
    GetReportSettings ............................................................................................... 39
EventTracker.Configs.RSS .................................................................................... 39
    GetRssFeeds ....................................................................................................... 40
EventTracker.Configs.Systems .............................................................................. 40
    GetAllSystemGroups ........................................................................................... 40
    GetAllSystems ..................................................................................................... 40
    GetSystem ........................................................................................................... 41
    GetSystemHierarchy ........................................................................................... 41
    GetSystemNames ................................................................................................. 42
EventTracker.Configs.Users .................................................................................. 42
    GetUser ............................................................................................................... 42
    GetUserBySAMAccountName .............................................................................. 42
    GetUserMenus ...................................................................................................... 43
    GetUsers ............................................................................................................. 43
EventTracker.Configs.Weights .............................................................................. 43
    GetCategoryWeightage ....................................................................................... 43
EventTracker.Dashboard.Inventory ..................................................................................................... 58
EventTracker.Dashboard.EAM ............................................................................................................. 55
EventTracker.Dashboard.ChangePolicy .............................................................................................. 48
GetUpdatesInstalled .......................................................................................................................... 59
GetApplicationInstalled ..................................................................................................................... 58
GetVolumeAnalysis ............................................................................................................................ 58
GetTopActivity .................................................................................................................................... 57
GetNewActivity ................................................................................................................................... 57
GetGraphActivity ................................................................................................................................ 56
GetDashboardSummary .................................................................................................................... 56
GetBreakupActivity ............................................................................................................................ 55
GetAlertChart ...................................................................................................................................... 46
GetAlertChartwithAcknowledgedStatus ............................................................................................ 47
GetTopAlerts ....................................................................................................................................... 47
GetTopAlertswithAcknowledgedStatus .............................................................................................. 47
EventTracker.Dashboard.ChangePolicy .............................................................................................. 48
GetBlacklistedObjects ........................................................................................................................ 48
GetChangePolicyDashboard .............................................................................................................. 48
GetChangedItems ............................................................................................................................ 48
GetChangedItemsGroupByPath ........................................................................................................... 49
GetSnapshot ....................................................................................................................................... 49
GetVulnerability .............................................................................................................................. 49
EventTracker.Dashboard.ConfigPolicy ............................................................................................... 50
GetComplianceResults_Assessment ................................................................................................... 50
GetComplianceResults_Benchmark .................................................................................................... 50
GetComplianceResults_RuleDetail .................................................................................................... 51
GetComplianceResults_RuleTitles ..................................................................................................... 51
GetConfigPolicyDashboard ............................................................................................................... 51
GetConfigPolicyDashboard_ResultId ............................................................................................... 51
GetConfigPolicyDashboard_ViewType ............................................................................................. 51
GetExecutedPolicySchedules_PolicyPage ......................................................................................... 52
GetExecutedPolicySchedules_TimeRange ......................................................................................... 52
getIntegrityViolations ......................................................................................................................... 52
GetIntegrityViolationsDetails ............................................................................................................ 52
GetPolicyNames ............................................................................................................................. 53
GetPolicySchedule ........................................................................................................................... 54
GetPolicyScheduleHistory .................................................................................................................. 54
GetPolicySchedules .......................................................................................................................... 55
GetPolicySchedules_PolicyPage ......................................................................................................... 55
EventTracker.Dashboard.EAM ............................................................................................................. 55
GetBreakupActivity ............................................................................................................................ 55
GetDashboardSummary .................................................................................................................... 56
GetFilters .......................................................................................................................................... 56
GetGraphActivity ............................................................................................................................ 56
GetNewActivity .............................................................................................................................. 57
GetTopActivity .............................................................................................................................. 57
GetVolumeAnalysis .......................................................................................................................... 58
EventTracker.Dashboard.Inventory ..................................................................................................... 58
GetApplicationInstalled ..................................................................................................................... 58
GetSystemInfo .................................................................................................................................... 58
GetUpdatesInstalled .......................................................................................................................... 59
<table>
<thead>
<tr>
<th>EventTracker.Reports</th>
<th>59</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetReportRequest</td>
<td>59</td>
</tr>
<tr>
<td>SetReportRequest</td>
<td>59</td>
</tr>
<tr>
<td>Samples</td>
<td>61</td>
</tr>
<tr>
<td>Sample #1</td>
<td>61</td>
</tr>
<tr>
<td>Sample #2</td>
<td>63</td>
</tr>
</tbody>
</table>
Preface

This preface provides an introduction to Prism and the EventTracker service. Please review the sections below and feel free to contact us with any questions you may have. As you are getting started with the EventTracker API, review the information below about Prism Microsystems, Inc. and our support services:

- About Prism Microsystems, Inc.
- Contact Prism Support

About Prism Microsystems, Inc.

Prism’s market leading family of enterprise log, change and configuration management solutions enable commercial enterprises, educational institutions and government organizations to meet compliance requirements with confidence, increase the security of their IT environments and reduce overall risk to their enterprise. Customers span multiple sectors including financial, communications, scientific, healthcare, banking and government. Prism solutions are currently deployed in production at over 850 global customers.

Prism Microsystems was formed in 1999 and is privately funded and held. Prism’s corporate headquarters are located in Columbia, Maryland in the Baltimore-Washington high tech corridor, with research and development facilities located in both Columbia and Bangalore, India. The company grew from the efforts of technologists who, while developing solutions in the telecom network management domain, discovered various problems in the areas of fault, configuration, accounting, performance and security (FCAPS). In the course of their activities early versions of EventTracker, Prism’s enterprise SIEM solution were created. Prism Microsystems was founded to develop, commercialize and support this solution and was a pioneer in the field of enterprise event log management.

As a service to the IT community, Prism Microsystems provides and maintains a free knowledge base of event information at [http://kb.prismmicrosys.com](http://kb.prismmicrosys.com). The Knowledge Base is a complimentary, searchable, vendor-neutral collection of cause-resolution information on over 20,000 events generated by hundreds of sources including Windows, Syslog, application specific logs and CISCO systems. The EventTracker Knowledge Base is the largest single repository of event knowledge in existence and has over 30,000 registered members. It is fast becoming the primary source of log and event management expertise on the Internet.
Contact Prism Support

Prism is committed to providing you with the most thorough support. Through online documentation, telephone help, and direct e-mail support support@eventtracker.com, Prism ensures that your questions will be answered in the fastest time possible.

EventTracker API Overview

The EventTracker API provides programmatic access to EventTracker services.

Developers can build custom applications, tools, and services to fetch EventTracker data to display in their own format or to use in their own applications. The API is based on open standards known collectively as ‘Web Services’, which include the Simple Object Access Protocol (SOAP), Web Services Definition Language (WSDL), and the XML Schema Definition language (XSD). These standards are supported by a wide range of development tools on a variety of platforms.

Using the EventTracker API, partners can access the following EventTracker features:

- EventTracker Configs
- EventTracker Dashboards
- EventTracker Reports
Services Architecture

From the Partner’s point of view, the system processes each EventTracker API request as shown in the following figure.

Step 1 - Receives an HTTPS Request

The partner application establishes a secure HTTP connection (using SSL encryption and ‘Windows’ authentication) with the EventTracker API Module.

Step 2 - Performs an EventTracker Function

The EventTracker manager performs a variety of functions, including Report API, Dashboard API, Config API, retrieving token information, retrieving account information on Systems, domains.

Step 3 - Returns an XML Report

After a function completes, the EventTracker manager returns a report or status message in XML format.
EventTracker User Account

The application must authenticate using EventTracker user account credentials (user name and password) as part of HTTP requests made to the EventTracker manager. For all web methods, an EventTracker account is required. If you need assistance with obtaining an EventTracker account, please contact your EventTracker support team.

Non-interactive users of EventTracker group can access the API to run web methods and view reports/configs.

Interactive users can access the EventTracker application. By default, all members of EventTracker user group are interactive users.

1. Log on to EventTracker Enterprise, click the Admin drop-down, and then click Users. User management window displays.
2. Clear the Interactive User check against the user you wish to make non-interactive. EventTracker displays the confirmation message box.
Decoding XML Reports

There are a number of ways to parse an XML file. Select the method which is most appropriate for your application and its users. EventTracker publishes DTDs for each report/configs on its Web site. For example, the URL to the report can be found at the URL shown below:


The URLs to current report DTDs are included with the web method descriptions in this document. There is a generic report returned by every web methods. Occasionally EventTracker updates the report DTDs. It is recommended that you request the most recent DTDs from the EventTracker platform to decode your reports. The URLs to the report DTDs are included in this user guide.

Some parts of the XML report may contain HTML tags or other special characters (such as accented letters). Therefore, many elements contain CDATA sections, which allow HTML tags to be included in the report. ‘High’ ASCII and other non-printable characters are escaped using question marks.

Authentication

The application must authenticate using EventTracker account credentials (user name and password) as part of the HTTP request. The credentials are transmitted using the ‘Windows Authentication Scheme’ over HTTPS.
Deploy EventTracker API

Summary

   You should have appropriate license to apply this patch. EventTracker displays an error message if the license verification fails.

   ![EventTracker Patch Installer](image)

   Figure 4

2. Create a Virtual Directory in IIS web server and name it as 'Prism.EventTracker.API'
3. Map the Virtual Directory to the 'EventTrackerAPI' folder
Set up Virtual Directory- Windows server 2003 (V6)

1. Open the IIS Manager

2. Right-click Default Web Site, point to New and then click Virtual Directory.

IIS Manager displays the Virtual Directory Creation Wizard.
3. Click **Next >**.

4. Type the alias for this Virtual Directory as ‘Prism.EventTracker.API’ in the **Alias** field.
   You can provide any alias name.
5. Click Next.
6. Click the Browse button and select the content folder that you want to publish on the Web site.
   (OR)
   Type the path of the folder in the Path field.

![Virtual Directory Creation Wizard](image)

Figure 8

7. Click Next > and assign Virtual Directory access permissions.
8. Click **Next >** and then click **Finish**.

IIS Manager displays the newly created Virtual Directory.
Figure 11
Configure Virtual Directory Properties


2. From the shortcut menu, choose **Properties**.

![Figure 12](image)

IIS Manager displays the Prism.EventTracker.API Properties window.

3. Select the **Directory browsing** check box, if not selected.

4. Select **Scripts only** option from the **Execute permissions** drop-down list.

5. Click the **Create** button.
IIS Manager,
  a. Updates and enables the **Application name** field with the name of the Virtual Directory
  b. Enables the **Configuration** button
  c. Enables the **Application pool** drop-down list

6. Click the **Directory Security** tab.
7. Click the **Edit** button under **Authentication and access** control.
   IIS Manager displays the Authentication Methods window.
8. Select the **Integrated Windows authentication** check box.
9. Click **OK**, click **Apply** and then click **OK** on the Prism.EventTracker.API Properties window.

10. Close the IIS Manager.
Browse the Virtual Directory

1. Open the Internet Explorer.
2. Type ‘http://localhost/Prism.EventTracker.API/’ in the address bar and then click GO.

If you access from a remote computer, replace ‘localhost’ with the name of the EventTracker API server.
Set up Virtual Directory- Windows Server 2008 (Version-7.0, 7.5)

1. Open the IIS Manager.
2. Right-click Default Web Site, click on Add Virtual Directory.

Add Virtual Directory window displays.
3. Type the alias for this Virtual Directory as ‘Prism.EventTracker.API’ in the Alias field.
   (OR)
   You can provide any alias name.
4. Click the Browse button and select the content folder that you want to publish on the Web site.
   (OR)
   Type the path of the folder in the Physical path field.
5. Click **OK**. Prism.EventTracker.API folder will be displayed. IIS Manager displays the newly created Virtual Directory.

6. Right-click the Virtual directory (i.e. in our example Prism.EventTracker.API) created, and then select **Convert to Application**.
7. Click **Content View**, select any file.
8. Click **Browse** hyperlink.
Prism.EventTracker.API window displays.
EventTracker Conventions

Supported Operations

- **Configs**: Read
- **Dashboard**: View
- **Reports**: Configure and View

Xml Structure

- **Configs/Dashboard**: Config.Dashboard.Structure.xml
- **Sample**: Config.Dashboard.Structure.Data.xml

- **Reports**: Reports.xml
- **Sample**: Reports.Data.xml

Xml Nodes/Attributes

- **Token No**: Unique No assigned to each report request. N/A for Configs/Dashboard
- **Status**: 0 = New, 2 = Process, 4 = Success, 5 = No Record Found, 6 = Failed
  - 4/6 is applicable for Configs/Dashboard
- 0/2/4/5/6 is applicable for Reports
- **Data**: Can contain Configs/Dashboard/Report Data
- **Error**: If any error comes during processing the request it will reflect here.

Web Service Files

- **Configs**: EventTracker.Configs.*.asmx
- **Dashboard**: EventTracker.Dashboard.*.asmx
- **Reports**: EventTracker.Reports.*.asmx
EventTracker Web Services

Provide non-interactive user name and password while calling web methods.

EventTracker.Configs.Alert


GetAlert

EventTracker.Configs.Alert.GetAlert (string username, string password, string AlertId);
Description: The GetAlert method is used to obtain configuration details of an Alert.
Parameters: AlertId
Alerts are identified by this unique id.
Nodes in XML Output: Alert Name, Alert Id, Product Name, Module Name, Weightage Id, Threshold Level, Status, Refine Rule Set, Filter Rule Set, Custom configuration, System(s)/System Group(s), and Alert Actions.
Comments: This method provides a simple way to collect configuration details of an Alert of your interest. To get Alert Ids, first execute the GetAlerts method and then locate the <AlertId> tag in the XML output.

GetAlerts

EventTracker.Configs.Alert.GetAlerts (string username, string password);
Description: The GetAlerts method is used to obtain configuration details of all Alerts.
Parameters: None
Nodes in XML Output: Alert Name, Alert Id, Product Name, Module Name, Weightage Id, Threshold Level, Status, Refine Rule Set, Filter Rule Set, Custom configuration, System(s)/System Group(s), and Alert Actions.
Comments: This method provides a simple way to collect configuration details of all Alerts.

EventTracker.Configs.Category


GetAllCategories

EventTracker.Configs.Category.GetAllCategories (string username, string password);
Description: The GetAllCategories method is used to obtain summary information of all Categories.

Parameters: None

Nodes in XML Output: Category Report Id and Name of the Category.
Category Report Id is unique to each Category.

Comments: This method provides a simple way to collect summary on all Categories that includes, Category Name and Category Report Id.

GetCategory

EventTracker.Configs.Category.GetAllCategories (string username, string password, string categoryId);

Description: The GetCategory method is used to obtain configuration details of a Category.

Parameters: Category Id

Nodes in XML Output: Category Name, Category Id, Product Name, Module Name, Report Id, Weightage Id, Category Description, Name of the creator, Name of the modifier, Group Mode, Active / Inactive status, Date created, Date modified, Refine Rule Set, and Filter Rule Set.

Comments: This method provides a simple way to collect details of a Category.

GetCategoryByGroup

EventTracker.Configs.Category.GetCategoryByGroup (string username, string password, string mode);

Description: The GetCategoryByGroup method is used to obtain configuration details of Categories by group.

Parameters: Mode.
   Accepted input: Category Group = 1
   Category = 2
   Report Group = 3
   Report = 4
   Report Chapter = 5

Comments: This method provides a simple way to collect details of Categories by group.

GetCategoryHierarchy

EventTracker.Configs.Category.GetCategoryHierarchy (string username, string password);

Description: The GetCategoryHierarchy method is used to obtain hierarchy of Categories.

Parameters: None

Nodes in XML Output: Relation Id, Name, Type Name (Category Group, Category), Cat Rep Id, Parent Id (0 = has no parent), and Type (0 = Category Group, 1 = Category).
**Comments:** This method provides a simple way to collect hierarchy of Categories.

### GetCompleteHierarchy


**Description:** The GetCompleteHierarchy method is used to obtain complete hierarchy of Categories.

**Parameters:** None

**Nodes in XML Output:** Parent Id, Parent Name, Cat Rep Id, Child Name, and Type.

**Comments:** This method provides a simple way to collect complete hierarchy of Categories.

### EventTracker.Configs.CollectionPoint


### GetCollectionPointCabStatus

*EventTracker.Configs.CollectionPoint.GetCollectionPointCabStatus*(string *username*, string *password*, string *collectionPointId*, string *cabStatusCode*);

**Description:** The GetCollectionPointCabStatus method is used to obtain status of all CAB files received from a Collection Point.

**Parameters:** Collection Point Id, CAB Status Code

- Collection Points are identified by this unique id.
- Valid CAB status:
  - Failed = 0
  - Success = 1
  - In Progress = 2
  - Queued = 3
  - Not Sent = 4
  - All = 5

**Nodes in XML Output:** Name of the CAB file, Time range of events stored in the CAB file, Collection Point Id, Name & IP address of Collection Point, Size of CAB file in KB, Date & Time when the transmission started, Time taken to complete the transmission, and Status of CAB file.

**Comments:** This method provides a simple way to collect information of CAB files received from Collection Point(s). To get Collection Point Ids, first execute the GetCollectionPointDetail method and then locate the <Client_Id> tag in the XML output.
GetCollectionPointDetail

EventTracker.Configs.CollectionPoint.GetCollectionPointDetail (string username, string password, string collectionPointIds);

Description: The GetCollectionPointDetail method is used to obtain configuration details of all Collection Points or one particular Collection Point.

Parameters: Collection Point Ids

Nodes in XML Output: Collection Point Id, Name & IP address of Collection Point, EventTracker Collection Point version, Path of the directory on EventTracker Enterprise server where Collection Point Archives are stored, Time Ticks of the last CAB file received by the Collection Point Manager, and Name of the last CAB file received.

Comments: This method provides a simple way to collect configuration details of Collection Point(s). ‘CollectionPointIds’ parameter is optional. Leave this field blank to view details of all Collection Point. To get Collection Point Ids, locate the <Client_Id> tag in the XML output. You can specify multiple ids by using comma separated values.

GetCollectionPointManagers

EventTracker.Configs.CollectionPoint.GetCollectionPointManagers (string username, string password);

Description: The GetCollectionPointManagers method is used to obtain details of Collection Point Managers.

Parameters: None

Nodes in XML Output: Unique Collection Point Manager Id, IP Address, Communication Port number, Status (1=Active), Description, and Encryption status (true = encryption enabled, false = encryption disabled).

Comments: This method provides a simple way to collect details of Collection Point Managers when executed from a Collection Point.

EventTracker.Configs.EAM


GetConfigs

EventTracker.Configs.EAM.GetConfigs (string username, string password);

Description: The GetConfigs method is used to obtain Enterprise Activity Monitoring Behavior settings.

Parameters: None

Nodes in XML Output: Product
Module
EA_BehaviourLearningPeriod
EA_SetLogSearchForEnterpriseActivity
EA_UserBehaviourCorrelationMonitoring
EA_SelectDNS
EA_SelectProcess
EA_UserActivityMonitoring
EA_SelectPurgeOldData
EA_BehaviourEventThreshold
EA_BehaviourLearningPeriodValue
EA_TopActivities
EA_UserBehaviourCorrelationThreshold
EA_UserEvtThreshold
EA_UserReportPurgeFreq
EA_EnterpriseInterval
DnsUrl
ProcessLib

Comments: This method provides a simple way to collect details of Enterprise Activity Monitoring Behavior settings.

GetCustomColumns

EventTracker.Configs.EAM.GetCustomColumns (string username, string password);
Description: The GetCustomColumns method is used to obtain details of custom columns configured.
Parameters: None
Nodes in XML Output: ID, Column Name, Column Type, Default, Key Value Splitter, Key Value Terminator, Column Alias, and Status.
Comments: This method provides a simple way to collect details of custom columns configured.

GetCustomRule

EventTracker.Configs.EAM.GetCustomRule (string username, string password, string ruleId);
Description: The GetCustomRule method is used to obtain details of a custom rule configured.
Parameters: Rule Id
Nodes in XML Output:
EntAct Custom Rule
Refine Rule Set
Event Rule
Event Details Id
Event Type
Log Type
Category
Event ID
User
Source
Description
Description Exception
EntAct Custom Rule Column
Custom Id
Rule Id
Column Name
Separator
Terminator
Display Name
Rule Id
Rule Name
Breakup Column Name
Breakup Separator
Breakup Terminator
User Name
Show In
Show For
Active
Weightage Id
PreDefined
Breakup Display Name.

Comments: This method provides a simple way to collect details of a custom rule configured.

GetCustomRules

EventTracker.Configs.EAM.GetCustomRules (string username, string password);
Description: The GetCustomRules method is used to obtain summary on all custom rules configured.
Parameters: None
Nodes in XML Output: Rule Id, Rule Name, Active (true = active, false = inactive), User Name, Weightage Id, pre defined (true = predefined, false = user-defined), and Weightage Name.
Comments: This method provides a simple way to collect summary on all custom rules configured.
GetCustomRulesByStatus

**EventTracker.Configs.EAM.GetCustomRulesByStatus (string username, string password, string statusCode);**

**Description:** The GetCustomRulesByStatus method is used to obtain summary on all custom rules by status.

**Parameters:** Status Code

**Nodes in XML Output:** Rule Id, Rule Name, Active (true = active, false = inactive), User Name, Weightage id, pre defined (true = predefined, false = user-defined), and Weightage Name.

**Comments:** This method provides a simple way to collect summary on all custom rules by status.

GetCustomRulesColumns

**EventTracker.Configs.EAM.GetCustomRulesColumns (string username, string password, string ruleId);**

**Description:** The GetCustomRulesColumns method is used to obtain summary on custom rule columns.

**Parameters:** Rule Id

**Nodes in XML Output:** Rule Name, Custom ID, Column Name, Separator, Terminator, and Rule Id.

**Comments:** This method provides a simple way to collect summary on custom rule columns.

GetCustomRulesEvents

**EventTracker.Configs.EAM.GetCustomRulesEvents (string username, string password, string ruleId);**

**Description:** The GetCustomRulesEvents method is used to obtain summary on custom rule events.

**Parameters:** Rule Id

**Sample Output:** ...

**Nodes in XML Output:** Rule Name, Event Details Id, Ref Id, Event Type, Log Type, Source, Description, Category, Event Id, Event User, Rule Type, Ref Type Identifier, Active (true = active, false = inactive).

**Comments:** This method provides a simple way to collect summary on custom rule events.

EventTracker.Configs.EventVault

GetCabInfo

\[ \text{EventTracker.Configs.EventVault.GetCabInfo (string username, string password, string fromdate, string todate);} \]

**Description:** The GetCabInfo method is used to obtain information on CAB files generated during a particular period on the Manager system.

**Parameters:** From Date, To Date

**Nodes in XML Output:** CAB Id, Name of the CAB file, Name of the MS Access MDB file, Date & Time range of events stored in the CAB file, Checksum size, Checksum value, Size of the CAB file, Count of events, Communication port number, Directory path where the CAB file is stored, Checksum type, and Name of the system.

**Comments:** This method provides a simple way to collect details of CAB files generated during a particular period on the Manager system.

GetCabInfoSystem

\[ \text{EventTracker.Configs.EventVault.GetCabInfoSystem (string username, string password, string fromdate, string todate, string systems);} \]

**Description:** The GetCabInfoSystem method is used to obtain summary on CAB files generated during a particular period on the Manager or remote Agent systems.

**Parameters:** From Date, To Date, Systems

**Nodes in XML Output:** Directory path and name of the CAB files.

**Comments:** This method provides a simple way to collect summary on CAB files generated during a particular period on the Manager or remote Agent systems.

EventTracker.Configs.KeyValuePair


GetKeyValuePair

\[ \text{EventTracker.Configs.KeyValuePair.GetKeyValuePair (string username, string password, string key);} \]

**Description:** The GetKeyValuePair method is used to obtain summary on configuration keys and values.

**Parameters:** key

**Nodes in XML Output:**

**Comments:** This method provides a simple way to collect summary on configuration keys and values.
GetKeyValuePairs

**EventTracker.Configs.KeyValuePair.GetKeyValuePairs (string username, string password);**

**Description:** The GetKeyValuePairs method is used to obtain summary on configuration keys and values.

**Parameters:** None

Comments: This method provides a simple way to collect summary on configuration keys and values.

### EventTracker.Configs.Manager


#### GetDLAConfiguration

*EventTracker.Configs.Manager.GetDLAConfiguration (string username, string password)*;

**Description:** The GetDLAConfiguration method is used to obtain summary DLA configuration settings.

**Parameters:** None

**Nodes in XML Output:** Log File Folder, Config File Name, LogFile Extension, Field Separator, and Log Type.

**Comments:** This method provides a simple way to collect summary on DLA configuration settings.

#### GetEmail

*EventTracker.Configs.Manager.GetEmail (string username, string password)*;

**Description:** The GetEmail method is used to obtain summary on e-mail configuration settings.

**Parameters:** None

**Nodes in XML Output:** Authentication (true = authentication enabled, false = authentication not enabled), SMTP Port, SMTP Server, From Email, User Name, and Password.

**Comments:** This method provides a simple way to collect summary on e-mail configuration settings.

#### GetNetFlowReceiver

*EventTracker.Configs.Manager.GetNetFlowReceiver (string username, string password)*;

**Description:** The GetNetFlowReceiver method is used to obtain summary on NetFlow Receiver Ports.

**Parameters:** None

**Nodes in XML Output:** Id, Port No, Drop Rate, Decode Packet (true / false), Record Binary (true / false).

**Comments:** This method provides a simple way to collect summary on NetFlow Receiver Ports.
GetSyslogPorts

```
EventTracker.Configs.Manager.GetSyslogPorts (string username, string password);
```

**Description:** The GetSyslogPorts method is used to obtain summary on Syslog Ports.

**Parameters:** None

**Nodes in XML Output:** Port Info Id, Port No, Comments, Port Info Identifier, Syslog Forward Enabled (true = enabled, false = disabled), Syslog Forward System, Syslog Forward Mode, and Syslog Forward Port.

**Comments:** This method provides a simple way to collect summary on Syslog Ports.

GetVCPPorts

```
EventTracker.Configs.Manager.GetVCPPorts (string username, string password);
```

**Description:** The GetVCPPorts method is used to obtain summary on VCP Ports.

**Parameters:** None

**Nodes in XML Output:** Port Info Id, Port No, Comments, Port Info Identifier, Syslog Forward Enabled (true = enabled, false = disabled), Syslog Forward System, Syslog Forward Mode, and Syslog Forward Port.

**Comments:** This method provides a simple way to collect summary on VCP Ports.

EventTracker.Config.ReceiverFilter


GetReceiverFilter

```
EventTracker.Config.ReceiverFilter.GetReceiverFilter (string username, string password, string filterId);
```

**Description:** The GetReceiverFilter method is used to obtain configuration details of a Receiver Filter.

**Parameters:** Filter Id

**Nodes in XML Output:** Product, Module, Filter Id, Filter Name, Description, Active, System Config, Refine Rule Set, and Filter Rule Set.

**Comments:** This method provides a simple way to collect configuration details of a Receiver Filter. To get Filter Ids, first execute the GetReceiverFilters method and then locate the `<Filter_Id>` tag in the XML output.

GetReceiverFilters

```
EventTracker.Config.ReceiverFilters.GetReceiverFilters (string username, string password);
```
**Description:** The GetReceiverFilters method is used to obtain summary information on Receiver Filters.

**Parameters:** None

**Nodes in XML Output:** Filter Id, Name, Description, System Mode, Systems, System Group, and Active (true = Active, false = Inactive).

**Comments:** This method provides a simple way to collect summary information on all configured Receiver Filters.

**EventTracker.Configs.ReportSettings**


**GetCostSavingAnalysis**

`EventTracker.Configs.ReportSettings.GetCostSavingAnalysis (string username, string password)`

**Description:** The GetCostSavingAnalysis method is used to obtain details of cost saving analysis settings.

**Parameters:** None

**Nodes in XML Output:** Cost Saving Analysis, Jr Admin Rate, Sr Admin Rate, Currency, Manual Analysis Cost, Manual Analysis, Time Taken, Analyst, Analysis Type, and Analysis Text.

**Comments:** This method provides a simple way to collect details of cost saving analysis settings.

**GetReportSettings**

**Description:** The GetReportSettings method is used to obtain details of report option settings.

**Parameters:** None

**Nodes in XML Output:** Report Options, Product, Module, and Default Report (true / false); Prompt To Publish Quick view Reports (true / false), Overwrite Domain User (true, false), Purge Frequency On Demand, Purge Frequency Scheduled, Attachment Size, Reports Path, and DNS URL.

**Comments:** This method provides a simple way to collect details of report option settings.

**EventTracker.Configs.RSS**

GetRssFeeds

*EventTracker.Configs.RSS.GetRssFeeds (string username, string password)*;

**Description:** The GetRssFeeds method is used to obtain information on RSS Feeds that includes Enterprise Feeds and My Feeds configured by all users.

**Parameters:** None

**Nodes in XML Output:** RSS Feed Id, Name of the RSS Feed, RSS Feed Type (0 = Enterprise, 1 = My Feeds), Description of the Feed, Name of the user who added the feed, Date & time when the feed was added, Status of the feed (true = Active, false = Inactive).

**Comments:** This method provides a simple way to collect information on RSS Feeds that includes Enterprise Feeds and My Feeds configured by all users.

EventTracker.Configs.Systems


GetAllSystemGroups

*EventTracker.Configs.Systems.GetAllSystemGroups (string username, string password)*;

**Description:** The GetAllSystemGroups method is used to obtain information on all enterprise domains.

**Parameters:** None

**Nodes in XML Output:** System Group Id, Name of the Group, Type of the group (2 = system group, 1 = system), whether Agent deployed manually (true / false), System type of the domain server, Communication port number, Log Time, Description of the group, IP address of the domain server, status (true = managed, false = unmanaged), Weightage, Active (true = active, false = inactive), Poll frequency, and Hardware details.

**Comments:** This method provides a simple way to collect information on all enterprise domains (system groups).

GetAllSystems

*EventTracker.Configs.Systems.GetAllSystems (string username, string password, string systemName, string exactMatch)*;

**Description:** The GetAllSystems method is used to obtain information on all systems or one particular system.

**Parameters:** System Name, Exact Match

**Nodes in XML Output:** System Group Id, Name, Type (1 = system, 2 = system group), Manual (true / false), Agentless (true / false), System Type (0 = Unknown, 2 = Windows 95, 3 = Windows 98, 13 = Windows ME, 4 = Windows NT - Workstation, 6 = Windows 2000 – Professional, 5= Windows NT – Server, 7 = Windows 2000 – Server, 8 = Windows 2000 –

**Comments:** This method provides a simple way to collect information on all enterprise domains (system groups). System Name is optional. If left blank it will fetch All Systems. Exact Match is optional. Default is false. Exact Match (True/False - Exact match of System Name) is required if System Name is present. For example: System Name = WEBDOC & Exact Match = left blank will fetch all system details that matches WEBDOC including DLA system instance. System Name = WEBDOC & Exact Match = WEBDOC1 will fetch details of the system queried for.

### GetSystem

*EventTracker.Configs.Systems.GetSystem (string username, string password, string systemId)*:

**Description:** The GetAllSystem method is used to obtain details of a system.

**Parameters:** System Id

**Nodes in XML Output:** Product, Module, System Type, Port, Weightage Id, Description, Install Directory, Version, IP Address, Components, System Config Type, Manual (type of Agent deployment), Active, Agentless (Agentless / Agent Based), Managed (managed, unmanaged), Log Time, System Group Type, System Name, and System Id.

**Comments:** This method provides a simple way to collect details of a system.

### GetSystemHierarchy


**Description:** The GetAllHierarchy method is used to obtain summary on hierarchy of all systems.

**Parameters:** None

**Nodes in XML Output:** Name of the system, Name of the parent group, System Type, System Group Id, and Parent Id.

**Comments:** This method provides a simple way to collect summary on hierarchy of all systems.
GetSystemNames

\texttt{EventTracker.Configs.Systems.GetSystemNames}\ (\texttt{string username, string password, string groupName});

\textbf{Description:} The GetSystemNames method is used to obtain name of the systems present in a queried system group.

\textbf{Parameters:} Group Name

\textbf{Nodes in XML Output:} System

\textbf{Comments:} This method provides a simple way to collect name of the systems present in a queried system group.

EventTracker.Configs.Users


GetUser

\texttt{EventTracker.Configs.Users.GetUser}\ (\texttt{string username, string password, string userId});

\textbf{Description:} The GetUser method is used to obtain summary on members EventTracker user group.

\textbf{Parameters:} User Id

\textbf{Nodes in XML Output:} Product, Module, Administrator (true /false), Status (true / false), First Name, Last Name, Initial, Modified By, SAM Account name, Member Of, Full Name, Modified Date, and User Id.

\textbf{Comments:} This method provides a simple way to collect summary on members of EventTracker user group by user id.

GetUserBySAMAccountName

\texttt{EventTracker.Configs.Users.GetUserBySAMAccountName}\ (\texttt{string username, string password, string samAccount});

\textbf{Description:} The GetUserBySAMAccountName method is used to obtain summary on members EventTracker user group.

\textbf{Parameters:} SAM Account

\textbf{Nodes in XML Output:} Product, Module, Administrator (true /false), Status (true / false), First Name, Last Name, Initial, Modified By, SAM Account name, Member Of, Full Name, Modified Date, and User Id.

\textbf{Comments:} This method provides a simple way to collect summary on members of EventTracker user group by SAM account name.
GetUserMenus

`EventTracker.Configs.Users.GetUserMenus (string username, string password);`

**Description:** The GetUserMenus method is used to obtain summary on menus available for members EventTracker user group.

**Parameters:** None

**Sample Output:** `...\XML\Configs\User.Menu.xml`

**Nodes in XML Output:** Menu Id, Menu Name, and URL.

**Comments:** This method provides a simple way to collect summary on menus available for members EventTracker user group.

**DTD:** `...\DTD\Configs\User.Menu.dtd`

**XSD:** `...\XSD\Configs\User.Menu.xsd`

GetUsers

`EventTracker.Configs.Users.GetUsers (string username, string password, string status);`

**Description:** The GetUsers method is used to obtain summary on members EventTracker user group.

**Parameters:** status

**Nodes in XML Output:** User Id, User First Name, User Last name, Status (true = Active, false = Inactive), Modified date, SAM Account name, User Full Name, and Administrator (true / false).

**Comments:** This method provides a simple way to collect summary on members EventTracker user group by status.

EventTracker.Configs.Weights


GetCategoryWeightage

`EventTracker.Configs.Weights.GetCategoryWeightage (string username, string password, string eventCategory);`

**Description:** The GetCategoryWeightage method is used to obtain summary on weightage assigned to event Categories.

**Parameters:** Event Category

**Nodes in XML Output:** Id, Weightage Value, Weightage Name, and Weightage Source (Category Name).

**Comments:** This method provides a simple way to collect summary on weightage assigned to event Categories.
Type exactly the Category for which you wish to collect weightage information. Otherwise, all Categories that meet the search criteria will be fetched. For example: ‘all error events’ will fetch summary on weightage for Juniper SBR: All error events, MSSQLServer: All error events, *All error events Categories.
*All error events will fetch summary on weightage for *All error events Category alone.

**GetEventIDWeightage**

```
EventTracker.Configs.Weights.GetEventIDWeightage (string username, string password, string eventId);
```

**Description:** The GetEventIDWeightage method is used to obtain summary on weightage assigned to event Ids.

**Parameters:** Event id

**Nodes in XML Output:** Id, Event ID, Weightage Id, Weightage Name, and Weightage Value.

**Comments:** This method provides a simple way to collect summary on weightage assigned to event Ids.

**GetEventSourceWeightage**

```
EventTracker.Configs.Weights.GetEventSourceWeightage (string username, string password, string eventSource);
```

**Description:** The GetEventSourceWeightage method is used to obtain summary on weightage assigned to event sources.

**Parameters:** Event Source

**Nodes in XML Output:** Id, Weightage Value, Weightage Name, and Weightage Source (event source).

**Comments:** This method provides a simple way to collect summary on weightage assigned to event sources.

**GetEventTypeWeightage**

```
EventTracker.Configs.Weights.GetEventTypeWeightage (string username, string password, string eventType);
```

**Description:** The GetEventTypeWeightage method is used to obtain summary on weightage assigned to event types.

**Parameters:** Event Type

**Nodes in XML Output:** Id, Weightage Value, Weightage Name, and Weightage Source.

**Comments:** This method provides a simple way to collect summary on weightage assigned to event types.
Event Type = Error, Warning, Information, Audit Success, Audit Failure, Success, Critical, andVerbose.

GetEventUserWeightage

`EventTracker.Configs.Weights.GetEventUserWeightage (string username, string password, string eventUser);`

**Description:** The GetEventUserWeightage method is used to obtain summary on weightage assigned to event users.

**Parameters:** Event User

**Nodes in XML Output:** Id, Weightage Value, Weightage Name, and Weightage Source.

**Comments:** This method provides a simple way to collect summary on weightage assigned to event users.

GetKeywordWeightage

`EventTracker.Configs.Weights.GetKeywordWeightage (string username, string password, string keyword);`

**Description:** The GetKeywordWeightage method is used to obtain summary on weightage assigned to keywords.

**Parameters:** Keyword

**Nodes in XML Output:** Keyword Id, Keyword, Weightage Id, Weightage Name, and Weightage Value.

**Comments:** This method provides a simple way to collect summary on weightage assigned to keywords.

GetLogTypeWeightage

`EventTracker.Configs.Weights.GetLogTypeWeightage (string username, string password, string logType);`

**Description:** The GetLogTypeWeightage method is used to obtain summary on weightage assigned to log types.

**Parameters:** Log Type

**Nodes in XML Output:** a

**Comments:** This method provides a simple way to collect summary on weightage assigned to log types.


GetSystemWeightage

EventTracker.Configs.Weights.GetSystemWeightage (string username, string password);
Description: The GetSystemWeightage method is used to obtain summary on weightage assigned to systems.
Parameters: None
Nodes in XML Output: Id, Weightage Value, Weightage Name, and Weightage Source.
Comments: This method provides a simple way to collect summary on weightage assigned to systems.

GetWeightageSettings

EventTracker.Configs.Weights.GetWeightageSettings (string username, string password);
Description: The GetWeightageSettings method is used to obtain summary on weightage settings.
Parameters: None
Nodes in XML Output: Id, Weightage Value, Weightage Name, and Weightage Source.
Comments: This method provides a simple way to collect summary on weightage settings.

EventTracker.Dashboard.Alert


GetAlertChart

EventTracker.Dashboard.Alert.GetAlertChart (string username, string password, string fromdate, string todate, string systems);
Description: The GetAlertChart method is used to obtain summary on Alert events generated on the managed systems.
Parameters: From Date, To Date, Systems
Nodes in XML Output: System Name, Alert Id, Alert Count, and Alert Name.
Comments: This method provides a simple way to collect summary on Alert events generated on the managed systems.
To collect Alert summary on multiple systems, use comma separator without space between system names. For example, systems = ELCDEMO,ELCTEST,LIMEGREEN
GetAlertChartwithAcknowledgedStatus

```csharp
EventTracker.Dashboard.Alert.GetAlertChart (string username, string password, string fromdate, string todate, string systems, Integer Acknowledged status);
```

**Description:** The GetAlertChartwithAcknowledgedStatus method is used to obtain acknowledge and unacknowledged summary on Alert events generated on the managed systems.

**Parameters:** From Date, To Date, Systems, Acknowledged status.

**Nodes in XML Output:** System Name, Alert Id, Alert Count, and Alert Name.

**Comments:** This method provides a simple way to collect Acknowledged and Unacknowledged status summary on Alert events generated on the managed systems.

GetTopAlerts

```csharp
EventTracker.Dashboard.Alert.GetTopAlerts (string username, string password, string fromdate, string todate, string systems, string sortBy);
```

**Description:** The GetTopAlerts method is used to obtain summary on top 20 Alerts generated on the managed systems.

**Parameters:** From Date, To Date, Systems, Sort By

**Nodes in XML Output:** Log Time, Event ID, System, Event Type, Source, User, Description, Risk Value, Risk Desc, Alert ID, Weightage, and Alert Name.

**Comments:** This method provides a simple way to collect summary on top 20 Alerts generated on the managed systems.

Sort By [0 = Log Time, 1 = Risk Value]

GetTopAlertswithAcknowledgedStatus

```csharp
EventTracker.Dashboard.Alert.GetTopAlerts (string username, string password, string fromdate, string todate, string systems, string sortBy, Integer Acknowledged status);
```

**Description:** GetTopAlertswithAcknowledgedStatus method is used to obtain Acknowledged summary on top 20 Alerts generated on the managed systems.

**Parameters:** From Date, To Date, Systems, Sort By, Acknowledged status.

**Nodes in XML Output:** Log Time, Event ID, System, Event Type, Source, User, Description, Risk Value, Risk Desc, Alert ID, Weightage, and Alert Name.

**Comments:** This method provides a simple way to collect summary on top 20 Alerts generated on the managed systems.

Sort By [0 = Log Time, 1 = Risk Value]
EventTracker.Dashboard.ChangePolicy


GetBlacklistedObjects

EventTracker.Dashboard.ChangePolicy.GetBlacklistedObjects (string username, string password, string resultId, string systemName, string categoryStatus);

Description: The GetBlacklistedObjects method is used to obtain details of blacklisted objects.

Parameters: Result Id, System Name, Category Status

Comments: This method provides a simple way to collect details of blacklisted objects.
Category Status [All = 0, Found = 1, Not Found = 2, Not Applicable = 3, Unknown = 4, Generic Error = 5, Failed To Read Snapshot = 6, Failed To Read Details = 7]
Result Id = 0

GetChangePolicyDashboard

EventTracker.Dashboard.ChangePolicy.GetChangePolicyDashboard (string username, string password, string systemGroup, string viewBy);

Description: The GetChangePolicyDashboard method is used to obtain details of Change Policy Dashboard.

Parameters: System Group, View By

Nodes in XML Output: Object Type –
Id (Snapshot Id), System Name, Cur Snapshot Ticks, Prev Snapshot Ticks, Cur Snapshot Time, Prev Snapshot Time, Files Added, Files Deleted, Files Modified, and Total Count.

Comments: This method provides a simple way to collect details of Change Policy Dashboard.
Group Name = TOONS
View By [Object Type = 0, Change Type = 1]

GetChangedItems

EventTracker.Dashboard.ChangePolicy.GetChangedItems (string username, string password, string systemName, string objectType, string objectStatus, string changeType, string snapshotId);

Description: The GetChangedItems method is used to obtain details of Changed items.

Parameters: System Group, View By

Comments: This method provides a simple way to collect details of Changed items.
Object Type [Both = 0, Files = 1, Registry = 2]
Object Status [All = 0, Modified = 1, Deleted = 2, Added = 3]
Change Type [All = 0, Authorized = 1, Unauthorized = 2, Configuration = 3, System = 4, Business Knowledge = 5]
GetChangedItemsGroupByPath

`EventTracker.Dashboard.ChangePolicy.GetChangedItemsGroupByPath (string username, string password, string systemName, string objectType, string objectStatus, string changeType, string snapshotId);`

**Description:** The GetChangedItemsGroupByPath method is used to obtain details of Changed items grouped by path.

**Parameters:** System Group, View By

**Comments:** This method provides a simple way to collect details of Changed items grouped by path.

Object Type [Both = 0, Files = 1, Registry = 2]
Object Status [All = 0, Modified = 1, Deleted = 2, Added = 3]
Change Type [All = 0, Authorized = 1, Unauthorized = 2, Configuration = 3, System = 4, Business Knowledge = 5]
Snapshot Id – Get Change Policy Dashboard - <Id>199</Id>

GetSnapshot

`EventTracker.Dashboard.ChangePolicy.GetSnapshot (string username, string password, string objectType, string itemId);`

**Description:** The GetSnapshot method is used to obtain details of Snapshots based on Object Type and Item Id.

**Parameters:** Object Type, Item Id


**Comments:** This method provides a simple way to collect details of Snapshots based on Object Type and Item Id.

Object Type [Files = 1, Registry = 2]
Item Id = Get Changed Items - <Id>98363</Id>

GetVulnerability

`EventTracker.Dashboard.ChangePolicy.GetVulnerability (string username, string password, string resultId, string systemName, string categoryStatus);`

**Description:** The GetVulnerability method is used to obtain details of Snapshots based on Object Type and Item Id.
Parameters: Result Id, System Name, Category Status
Comments: This method provides a simple way to collect details of Snapshots based on Object Type and Item Id.
Category Status [All = 0, Found = 1, Not Found = 2, Not Applicable = 3, Unknown = 4, Generic Error = 5, Failed To Read Snapshot = 6, Failed To Read Details = 7]
Result Id = 0

EventTracker.Dashboard.ConfigPolicy

GetComplianceResults_Assessment

EventTracker.Dashboard.ConfigPolicy.GetComplianceResults_Assessment (string username, string password, string TestResultId);
Description: The GetComplianceResults_Assessment method is used to obtain details of FDCC Assessment.
Parameters: Test Result Id
Nodes in XML Output: Test Result Id, Benchmark Id, Id Text, End Time, test system, Organization, Target, Platform, Score, Result File Path, Result File Name, Id, Result Id, Session Id, Performed At, System Name, Policy Name, Policy Description, Policy Schedule Name, Policy Schedule Freq, Result, Result String, Num File Changes, Num Reg Changes, Benchmark Class, and Test Result Id.
Comments: This method provides a simple way to collect details of FDCC Assessment.
Test Result Id = Get Executed Policy Schedules

GetComplianceResults_Benchmark

EventTracker.Dashboard.ConfigPolicy.GetComplianceResults_Benchmark (string username, string password, string TestResultId);
Description: The GetComplianceResults_Benchmark method is used to obtain benchmark details of FDCC Assessment.
Parameters: Test Result Id
Nodes in XML Output: Id, File Path, File Name, File Type, Format, Title, Description, Notice, Profile, Version, Platform, Publisher, OS Type, Result File Type, Publish Date, Publish Status, Class, Imported On, and Comments.
Comments: This method provides a simple way to collect benchmark details of FDCC Assessment.
Test Result Id = Get Executed Policy Schedules
GetComplianceResults_RuleDetail

EventTracker.Dashboard.ConfigPolicy.GetComplianceResults_RuleDetail(string username, string password, string ruleResultId);

**Description:** The GetComplianceResults_RuleDetail method is used to obtain rule details of FDCC Assessment.

**Parameters:** Rule Result Id

**Nodes in XML Output:** Rule Result Id, Test Result Id, Rule Id, Rule Title, Rule Description, Eval Time, Ident, Ident System, Reference, GP Reference, and Result.

**Comments:** This method provides a simple way to collect rule details of FDCC Assessment.

rule Result Id = Get FDCC Compliance Results Rule Titles

GetComplianceResults_RuleTitles

EventTracker.Dashboard.ConfigPolicy.GetComplianceResults_RuleTitles(string username, string password, string TestResultId);

**Description:** The GetComplianceResults_RuleTitles method is used to obtain rule titles of FDCC Assessment.

**Parameters:** Test Result Id

**Nodes in XML Output:** Rule Id, Rule Result Id, Rule Title, Result, and Ident (CCE Id).

**Comments:** This method provides a simple way to collect rule titles of FDCC Assessment.

test Result Id = Get Executed Policy Schedules –

GetConfigPolicyDashboard

EventTracker.Dashboard.ConfigPolicy.GetConfigPolicyDashboard(string username, string password);

**Description:** The GetConfigPolicyDashboard method is used to obtain Config Policy Dashboard details.

**Parameters:** None

**Nodes in XML Output:** Executed, Result Id, Policy Name, Policy Description, System Name, File Changes, Registry Changes, Integrity Violation, Frequency, Benchmark Class, Test Result Id, Benchmark Id, and Comparison Time.

**Comments:** This method provides a simple way to collect Config Policy Dashboard details.

GetConfigPolicyDashboard_ResultId

EventTracker.Dashboard.ConfigPolicy.GetConfigPolicyDashboard_ResultId(string username, string password, string resultId);

**Description:** The GetConfigPolicyDashboard_ResultId method is used to obtain FDCC Assessment test details by result id.

**Parameters:** None
Nodes in XML Output: Executed, Result Id, Policy Name, Policy Title, Policy Description, System Name, Integrity Violation, Comparison Time, Frequency, Benchmark Class, Test Result Id, Benchmark Id, Score, File Path, and File Name.

Comments: This method provides a simple way to collect FDCC Assessment test details by result id.
Result Id = Get Config Policy Dashboard –

GetConfigPolicyDashboard_ViewType

EventTracker.Dashboard.ConfigPolicy.GetConfigPolicyDashboard_ViewType (string username, string password, string startTime, string endTime, string viewType, string policyPage);

Description: The GetConfigPolicyDashboard_ViewType method is used to obtain FDCC Assessment test details by result id.

Parameters: User Name, Password, Start Time, End Time, View Type, Policy Page

Nodes in XML Output: Config Policy –
Executed, Result Id, Policy Name, Policy Title, System Name, File Changes, Registry Changes, Comparison Time, Frequency, Benchmark Class, Test Result Id, Result, ID, Benchmark Id, Result File Path, Passed, Failed, Exceptions, and Deviations.

Comments: This method provides a simple way to collect FDCC Assessment test details by result id.
Config Policy View Type [Success = 1 [Config Policy / Change Policy], Integrity Violations = 2 [Change Policy], Exceptions = 3 [Change Policy], Fail = 4 [Config Policy]]
Policy Page [Config Policy = 1, Change Policy = 2]

GetExecutedPolicySchedules_PolicyPage

EventTracker.Dashboard.ConfigPolicy.GetExecutedPolicySchedules_PolicyPage (string username, string password, string policyPage);

Description: The GetExecutedPolicySchedules_PolicyPage method is used to obtain details of executed policy schedules (Change & Config).

Parameters: Policy Page

Nodes in XML Output: Executed, Result Id, Policy Name, Policy Title, System Name, File Changes, Registry Changes, Comparison Time, Frequency, Benchmark Class, Test Result Id, Result, Passed, Failed Exceptions, and Deviations.

Comments: This method provides a simple way to collect details of executed policy schedules (Change & Config).
Policy Page [Config Policy = 1, Change Policy = 2]
GetExecutedPolicySchedules_TimeRange

EventTracker.Dashboard.ConfigPolicy.GetExecutedPolicySchedules_TimeRange (string username, string password, string policyPage, string fromTime, string toTime, string searchTitle);

Description: The GetExecutedPolicySchedules_TimeRange method is used to obtain details of executed policy schedules (Change & Config) within a specified time range.

Parameters: User Name, Password, Policy Page, From Time, To Time, Search Title

Nodes in XML Output: Executed, Result Id, Policy Name, Policy Title, System Name, File Changes, Registry Changes, Comparison Time, Frequency, Benchmark Class, Test Result Id, Result, Passed, Failed Exceptions, and Deviations.

Comments: This method provides a simple way to collect details of executed policy schedules (Change & Config) within a specified time range.
Policy Page [Config Policy = 1, Change Policy = 2]
Search Title = Get Executed Policy Schedules –

GetIntegrityViolations

EventTracker.Dashboard.ConfigPolicy.GetIntegrityViolations (string username, string password, string resultId);

Description: The GetIntegrityViolations method is used to obtain summary on integrity violations.

Parameters: Result Id

Nodes in XML Output: Object Type, Master Result Id, Id, and Item.

Comments: This method provides a simple way to collect summary on integrity violations.
Result Id = Get Executed Policy Schedules Policy Page - <Result_Id>4b8f7e6d</Result_Id> It's only for Change Policy

GetIntegrityViolationsDetails

EventTracker.Dashboard.ConfigPolicy.GetIntegrityViolationsDetails (string username, string password, string objectType, int violationId);

Description: The GetIntegrityViolationsDetails method is used to obtain summary on policies (Change & Config).

Parameters: Object Type, Violation Id

Nodes in XML Output: Policy Values Size, Policy Values Version, Created At Policy, Policy Values Creation Time, Policy Values Checksum, Actual Values Size, Actual Values Version, Created At Actual, Actual Values Creation Time, Actual Values Checksum, Description, and Item Location.

Comments: This method provides a simple way to collect summary on policies (Change & Config).
Object Type = Get Integrity Violations [Files = 1, Registry = 2] <ObjectType>1</ObjectType>
Violation Id = Get Integrity Violations - <Id>696</Id>
GetPolicyNames

`EventTracker.Dashboard.ConfigPolicy.GetPolicyNames (string username, string password, string policyPage);`

**Description:** The GetPolicyNames method is used to obtain summary on policies (Change & Config).

**Parameters:** Policy Page

**Nodes in XML Output:** Config Policy - Id, Name

**Comments:** This method provides a simple way to collect summary on policies (Change & Config).

Policy Page [Config Policy = 1, Change Policy = 2]

GetPolicySchedule

`EventTracker.Dashboard.ConfigPolicy.GetPolicySchedule (string username, string password, string scheduleId, string policyPage);`

**Description:** The GetPolicySchedule method is used to obtain details of policy schedules (Change & Config) based on schedule id and policy type.

**Parameters:** Schedule Id, Policy Page

**Nodes in XML Output:** Change Policy - Schedule Id, Policy Title, Policy Description, Sch Ticks, Systems, Sch Freq, Benchmark Id, Active, Sch Type, Policy Name, Schedule Time, Added By, Updated By, Added At, and Updated At.

**Comments:** This method provides a simple way to collect details of policy schedules (Change & Config) based on schedule id and policy type.

Schedule Id = Get Policy Schedules - Policy Page [Config Policy = 1, Change Policy = 2]

GetPolicyScheduleHistory

`EventTracker.Dashboard.ConfigPolicy.GetPolicyScheduleHistory (string username, string password, string policyTitle);`

**Description:** The GetPolicyScheduleHistory method is used to obtain details of execution history of policy schedules (Change and Config).

**Parameters:** Policy Title

**Nodes in XML Output:** Config Policy - Id, Result Id, Session Id, Performed At, System Name, Policy Name, Policy Description, Policy Schedule Name, Policy Schedule Freq, Result, Result String, Num File Changes, Num Reg Changes, Benchmark Class, TestResult_Id, Test_Result_Id, Benchmark Id, Id Text, End Time, test system, Organization, Target, Platform, Score, Result File Path, and Result File Name.

**Comments:** This method provides a simple way to collect details of execution history of policy schedules (Change and Config).

Policy Page [ConfigPolicy = 1, ChangePolicy = 2]
GetPolicySchedules

`EventTracker.Dashboard.ConfigPolicy.GetPolicySchedules (string username, string password);`

**Description:** The GetPolicySchedules method is used to obtain details of all policy schedules (Change and Config).

**Parameters:** None

**Nodes in XML Output:** Config Policy & Change Policy -
Id, Title, Description, Benchmark Id, Frequency, Systems, Schedule Time, Active (true = active, false = inactive), Added By, Updated By, Added At, Updated At, Schedule Type, Schedule Date, and Policy Name.

**Comments:** This method provides a simple way to collect details of all policy schedules (Change and Config).
Policy Page [ConfigPolicy = 1, ChangePolicy = 2]

GetPolicySchedules_PolicyPage

`EventTracker.Dashboard.ConfigPolicy.GetPolicySchedules_PolicyPage (string username, string password, string policyPage);`

**Description:** The GetPolicySchedules_PolicyPage method is used to obtain details of policy schedules (Change and Config).

**Parameters:** Policy Page

**Nodes in XML Output:** Config Policy –
Id, Title, Description, Benchmark Id, Frequency, Systems, Schedule Time, Active (true = active, false = inactive), Added By, Updated By, Added At, Updated At, Schedule Type, Schedule Date, and Policy Name.

**Comments:** This method provides a simple way to collect details of policy schedules (Change and Config).
Policy Page [ConfigPolicy = 1, ChangePolicy = 2]

EventTracker.Dashboard.EAM

**URL:** http://Server/VirutalDirectory/EventTracker.Dashboard.EAM.asmx

GetBreakupActivity

`EventTracker.Dashboard.EAM.GetBreakupActivity (string username, string password, string fromDate, string toDate, string typeofActivity, string searchString, string ruleName);`

**Description:** The GetBreakupActivity method is used to obtain Enterprise Activities break up details.

**Parameters:** From Date, To Date, Type of Activity Code, Search String, Rule Name

**Comments:** This method provides a simple way to collect Enterprise Activities break up details.
Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom] 

Search String = GetTopActivity TopActivities/TopActivity/ActivityType Node/Node Value


ruleName = is applicable only if Activity Type is 13

GetDashboardSummary

EventTracker.Dashboard.EAM.GetDashboardSummary (string username, string password, string fromDate, string toDate, string typeofActivity, string ruleName);

Description: The GetDashboardSummary method is used to obtain summary on dashboards.

Parameters: From Date, To Date, Type of Activity Code, Search String, Rule Name

Nodes in XML Output: Group Value, Count.

Comments: This method provides a simple way to collect summary on dashboards.

Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom]

ruleName = is applicable only if Activity Type is 13

GetFilters

EventTracker.Dashboard.EAM.GetFilters (string username, string password, string typeofActivity);

Description: The GetFilters method is used to obtain summary on Enterprise Activities Monitoring filters.

Parameters: Type of Activity Code

Nodes in XML Output: Filter Name, Exclude Type

Comments: This method provides a simple way to collect summary on Enterprise Activities Monitoring filters.

Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom]

GetGraphActivity

EventTracker.Dashboard.EAM.GetGraphActivity (string username, string password, string fromDate, string toDate, string typeofActivity, string searchString, string ruleName);
Description: The GetGraphActivity method is used to obtain summary on Enterprise Graph Activities.

Parameters: From Date, To Date, Type of Activity Code, Search String, Group Value, Rule Name

Nodes in XML Output: Intervals, Event Count

Comments: This method provides a simple way to collect summary on Enterprise Graph Activities.

Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom]

searchString = GetTopActivity TopActivities/TopActivity/ActivityType Node/Node Value


ruleName = is applicable only if Activity Type is 13

GetNewActivity

EventTracker.Dashboard.EAM.GetNewActivity (string username, string password, string typeofActivity, string ruleName);

Description: The GetNewActivity method is used to obtain summary on new activities.

Parameters: Type of Activity Code, Rule Name

Comments: This method provides a simple way to collect summary on new activities.

Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom]

ruleName = is applicable only if Activity Type is 13

GetTopActivity

EventTracker.Dashboard.EAM.GetTopActivity (string username, string password, string(fromDate, string toDate, string typeofActivity, string sortBy, string ruleName);

Description: The GetTopActivity method is used to obtain summary on top activities.

Parameters: From Date, To Date, Type of Activity Code, Search String, Sort By, Rule Name

Comments: This method provides a simple way to collect summary on top activities.

Activity Type [1 = User, 2 = Admin, 3 = EventId, 4 = IP, 5 = System, 6 = LoginFailure, 7 = Process, 8 = USB, 9 = RunawayProcess, 10 = Software, 11 = Application, 12 = NetworkAddress, 13 = Custom]

sortBy [0 = Count, 1 = Name]

ruleName = is applicable only if Activity Type is 13
GetVolumeAnalysis

**EventTracker.Dashboard.EAM.GetVolumeAnalysis (string username, string password, string ruleName);**

**Description:** The GetVolumeAnalysis method is used to obtain summary on event volume analysis.

**Parameters:** Rule Name

**Nodes in XML Output:** Activity, Distinct, and Total Count.

**Comments:** This method provides a simple way to collect summary on event volume analysis.

ruleName = is applicable only if Activity Type is 13

EventTracker.Dashboard.Inventory

URL: http://Server/VirutalDirectory/EventTracker.Dashboard.Inventory.asmx

GetApplicationInstalled

**EventTracker.Dashboard.Inventory.GetApplicationInstalled (string username, string password, string systemName);**

**Description:** The GetApplicationInstalled method is used to obtain details of applications installed on a system.

**Parameters:** System Name

**Nodes in XML Output:** Estimated Size, Install Date, Install Location, Install Source, Language, Readme, Uninstall String, Uninstall Path, Modify Path, Name, Description, Display Version, Version Major, Version Minor, Help Link, Publisher, URL Info About, URL Update Info, Help Telephone, Contact, Comments, IsWindowsInstaller, IsSystemComponent, CPE Id, and Additional Info.

**Comments:** This method provides a simple way to collect details of applications installed on a system.

GetSystemInfo

**EventTracker.Dashboard.Inventory.GetSystemInfo (string username, string password, string systemName);**

**Description:** The GetSystemInfo method is used to obtain hardware and other details of a system.

**Parameters:** System Name

**Nodes in XML Output:** IP Address, FQDN, Page Size, Number Of Processors, processor architecture, processor identifier, OS Id, OS suite mask, total physical memory, total virtual memory, total page file size, system directory path, windows directory path, root drive, OS Name OS Major Version, OS Minor Version, OS CSD Version, OS CSD Major, OS CSD Minor, OS
Build Number, OS product type, processor architecture, CPE Id, OVAL Id, Update URL, Additional Info, and Domain.

**Comments:** This method provides a simple way to collect hardware and other details of a system.

### GetUpdatesInstalled

```
EventTracker.Dashboard.Inventory.GetUpdatesInstalled (string username, string password, string systemName);
```

**Description:** The GetUpdatesInstalled method is used to obtain details of updates, hotfixes and patches applied on a system.

**Parameters:**
- System Name

**Nodes in XML Output:** Install Date, Installed By; Uninstall String, Description, Display Version, Package Name, Package Version, Publisher, Publishing Group, Installer Name, Installer Version, Help Link, and URL Info About, CVE Id, Additional Info, IsInAddRemove, and Update Type.

**Comments:** This method provides a simple way to collect details of updates, hotfixes and patches applied on a system.

### EventTracker.Reports


### GetReportRequest

```
EventTracker.Reports.GetReportRequest (string username, string password, string TokenNo);
```

**Description:** The GetReportRequest method is used to obtain details of report request.

**Parameters:** TokenNo

**Nodes in XML Output:** ReportAPI TokenNo="21" Status="4", Summary Group="Computer" Count="5", Group Value, Event IDs, Event Count, and Event ID.

**Comments:** This method provides a simple way to collect details of report request.

### SetReportRequest

```
EventTracker.Reports.SetReportRequest (string username, string password, string ReportType, string Chapter, string FromTime, string ToTime, string Systems, string Category);
```

**Description:** The SetReportRequest method is used to submit report request.

**Parameters:** Report Type, Chapter, From Time, To Time, Systems, Category

**Nodes in XML Output:** <ReportAPI TokenNo="16" Status="0">
- Status [0 = Initialize, 2 = Processing, 4 = Success, 5 = No Record Found, 6 = Failed, 8 = Cancelled]

**Comments:** This method provides a simple way to submit report request.
Logs - Summary [ReportType = 100, Chapter = 0]
Logs - Detail [ReportType = 100, Chapter = 1]
FromTime = 04/04/2010 09:00:00 AM. From Time is optional. If not present then it will consider previous day.
ToTime = 04/04/2010 09:00:00 PM. To Time is optional. If not present then it will consider Current date time.
Systems = Sys1, Sys2. Systems parameter is optional. If not present then it will consider all systems. Multiple systems can be supplied with comma separated values.
Category = *All information events. Category parameter is optional. If not present then it will consider all events.
Samples
Sample #1

1. Create a sample Web application.

2. Open the 'Add Web Reference'. Provide the URL of the web service hosting server. Example: http://[ServerName]/Prism.EventTracker.API/

3. A list of available Web services is displayed. Select the Web service you wish to use. Example: http://[ServerName]/Prism.EventTracker.API/EventTracker.Configs.Alert.asmx. This Web service provides Alert configuration information.
4. Following is the C# code snippet to call the web service method ‘GetAlerts’. Insert the below code in the code behind file of a web page.

```csharp
//Create a web service instance
EvenTrackerAlertAPI.EventTrackerConfigsAlert configAlert = new
EvenTrackerAlertAPI.EventTrackerConfigsAlert();
//call the web service method to get alerts list. The output is an xml string.
string xml = configAlert.GetAlerts("username", "password");
//create xml document instance.
XmlDocument xmlDoc = new XmlDocument();
//load the output xml into the xml document.
xmlDoc.LoadXml(xml);
//Get the data node in the xml.
string xmlData = xmlDoc.SelectSingleNode("ReportAPI/Data").InnerXml;
//using a dataset read the xml data.
DataSet ds = new DataSet("Alerts");
ds.ReadXml(new StringReader(xmlData));
//bind to a data grid.
grdAlerts.DataSource = ds;
grdAlerts.DataBind();
```
Sample #2

Following sample function demonstrates calling the web service API and getting back the resultant XML string.

**Parameters:**
strAsmxUrl: URL of the web service's asmx page
strSOAPAction: SOAP action for the asmx page
strXmlBody: XML body of the request

**Return value:**
Resultant XML string returned by the service

```cpp
CString PostWebService(CString strAsmxUrl, CString strSOAPAction, CString strXmlBody)
{
    try
    {
        CString strParams;
        XmlHTTPRequest pXmlReq(CLSID_ServerXMLHTTP60);
        XmlProcessor *pProcessor = XmlProcessor::Instance();
        XmlDocument pXmlDoc(CLSID_DOMDocument60);
        _bstr_t bstrXmlBody(strXmlBody);
        _bstr_t bstrAsmxUrl(strAsmxUrl);
```
Sample code to call the function
{
CString strAsmxUrl,strSOAPAction,strXmlBody,strOut;
CString strServer;
CString strUserName,strPassword;
strServer.Format("APIServer");
strUserName.Format("Domain\user1");
strPassword.Format("Password");
strAsmxUrl.Format("http://%s/Prism.EventTracker.API/EventTracker.Dashboard.ChangePolicy.asmxml\",strServer);
strXmlBody.Format("<?xml version="1.0" encoding="utf-8"?>");
strXmlBody+="<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
>";
strXmlBody+="<username>" + strUserName + "</username>";
strXmlBody+="<password>" + strPassword + "</password>";
strXmlBody+="<systemGroup>toons</systemGroup>";
strXmlBody+="<viewBy>" + strViewBy + "</viewBy>";
strXmlBody+="</GetChangePolicyDashboard>
</soap:Body>
</soap:Envelope>

try
{
  strOut = PostWebService(strAsmxUrl, strSOAPAction, strXmlBody);
  strOut.Replace("&", ";");
  strOut.Replace("&lt;", "");
  strOut.Replace("&apos;", ";");
  strOut.Replace("&quot;", ";");
  m_eOutputXML.SetWindowText(strOut);
  _bstr_t bstrOut(strOut);
  g_pOutXml->loadXML(bstrOut);
  FillXMLTree(g_pOutXml,TVI_ROOT);
}

catch(_com_error &e)
{
  CString strErr;
  _bstr_t bstrSource(e.Source());
  _bstr_t bstrDescription(e.Description());
  strErr.Format("COM exception occurred while sending request, Code: %08lx (%s), Source: %s, Desc: %s", e.Error(), e.ErrorMessage(), (LPCSTR) bstrSource, (LPCSTR) bstrDescription);
  AfxMessageBox(strErr);
}

catch(...) {
  CString strErr;
  strErr.Format("Unknown Exception occurred while sending request");
  AfxMessageBox(strErr);
}