# Purpose

The FAS.Configuration.AutoDiscovery solution consists of a single assembly API and a client application that can be used if your application does not integrate using the API. This document will cover how the AutoDiscover process interacts with the FAS.Configuration service and client applications, and how client applications must configure their AutoDiscovery configuration in order to retrieve their configurations.

# Scope

The AutoDiscovery solution depends on the presence of DNS entries in order to function. This document does not cover the setup or management of the DNS portion of AutoDiscovery. This document focuses on client configurations, but only covers applications integrated with the AutoDiscovery API.

# AutoDiscovery Process

The AutoDiscovery solution acts as a middle man for an application that needs to retrieve its application configuration file. Application configuration information is copied stored in the Forensic Advantage Configuration database table. The configuration entries are identified using a GUID that is the assembly GUID of the executing application or service. Provide the AutoDiscovery API with this application GUID to retrieve the proper configuration information from the Configuration database.

Applications and services integrated with the AutoDiscovery API retrieve their configurations before the application or service starts. The initial call to the AutoDiscovery API starts a process where the AutoDiscovery code checks for the presence of an AutoDiscovery.config file in the root of the application execution folder. This file is an XML configuration file, similar to a .Net Application Configuration file. The values in the AutoDiscovery.config file drive the AutoDiscovery process.

# AutoDiscovery Configuration Elements

The AutoDiscovery.config must first register a custom configuration section used by the AutoDiscovery process. The first element in the AutoDiscovery.config file must be the <configSections> element that contains the autoDiscoveryConfiguration section.

<configSections>

<section name=**"autoDiscoveryConfiguration"**

           type=**"FAS.Configuration.AutoDiscovery.Common.Configuration.Sections.ServiceApplicationConfigurationSection, FAS.Configuration.AutoDiscovery"** />

</configSections>

After registering the custom configuration section, the configuration file must contain the **autoDiscoveryConfiguration** element. This element contains a single root element named **<applications>**. Within the **<applications>** element is where individual application identities are coupled to a service that contains their configuration information. The AutoDiscovery client application can work with multiple application configurations simultaneously. Applications that integrate with AutoDiscovery using the API will only have a single application defined in the **<applications>** element.

<autoDiscoveryConfiguration>

<applications>

<application applicationId=**"F6327FCB-06DD-4BBC-B50F-C9EF942AB1F5"**

path=**"C:\Hosts\FAS.Configuration.WindowsService.exe"**

serviceName=**"FAS.Configuration"**

defaultContext=**"STAGE"**>

<service name=**"ConfigurationService"** domain=**"dev.tcsc.com"**>

<protocols>

<protocol key=**"net.tcp"** value=**"\_tcp"** />

</protocols>

<security spnIdentity=**"DPR\Configuration"** />

</service>

</application>

</applications>

</autoDiscoveryConfiguration>

An application registered in the AutoDiscovery.config file must have an applicationId. The application id is a Global Unique Identifier (GUID) that is defined in the assembly code. Though this value can be retrieved using command line tools, the value is static and does not change across product releases. Table 1 - Application Identity Mapping below should be accurate for the life of the applications.

Table - Application Identity Mapping

| Application | Id |
| --- | --- |
| FAS.Sentinel.Serialization.Administration | 5255EA83-6E65-4EF2-A523-99EA038CF2A8 |
| FAS.Sentinel.Serialization.Client | F6327FCB-06DD-4BBC-B50F-C9EF942AB1F5 |
| FAS.Sentinel.Serialization.WindowsService | 1166DC87-154B-4186-B25B-DEE56895767B |
| FAS.Sentinel.Serialization.RequestQueueService | E9DEFABA-A7CC-4457-8E26-91268D5CEE14 |
| FAS.Sentinel.Serialization.Console | 42109C80-DE48-454E-878B-B4F3AAD2A8EC |
| FAS.Sentinel.Serialization.RequestQueueConsole | 042E84DB-84DE-4D31-86A2-A178C6ECA984 |

An application element may contain a **defaultContext** attribute. This attribute defines the context to which the application or service should automatically connect. If the defaultContext is not defined, the user is prompted to choose the service context from a list of available Contexts in the environment. When using the AutoDiscovery API with a Windows Service, there is no opportunity to query a user interface for the service Context. The **defaultContext** should always be provided.

The **serviceName** attribute is used by the AutoDiscovery client application to properly identify a running service in the Windows Service Manager. The **serviceName** is not used by applications that interact with AutoDiscovery using API.

In addition to the applicationId, an application configuration must contain a **service** element that contains configuration information for the service that hosts the applications configuration data. (Do not confuse this with the **serviceName** attribute.) The **service** element must contain a **Name** attribute, which is the name of the service endpoint used to retrieve its configuration. The **domain** attribute should contain the domain name in which the service is hosted. The **protocols** element contains a list of protocols that can be used by the service. Only one protocol should be added to protocols collection that is the protocol used by the configuration service. Finally, a security element can be added, but is not required, if the configuration service endpoint is hosted using an account with an associated SPN.

The following tables describe the various elements and attributes of the AutoDiscovery configuration file.

## Application Element

| Entry | Description |
| --- | --- |
| ApplicationId (Required) | A GUID representing the applications Assembly GUID |
| Name | The assembly name of the application or service |
| DefaultContext | The name of the default configuration context. For example, if an application should start using configuration values for a Production environment, the default context would equal “PROD”. |
| Path | The path to the assembly executable. This value is not required for applications that use the AutoDiscovery API because the value is assigned programmatically. |
| Service | The container element for the service definition |
| ServiceName | If this configuration is for a Windows Service, the serviceName element should be the name of the service as listed in the Services Console. |
| UseExistingAppConfig | This flag can be used to override the AutoDiscovery process. If this value is set to true, AutoDiscovery will exit without attempting to download the configuration from the FAS.Configurtion service. |

## Service Element

| Entry | Description |
| --- | --- |
| Name (Required) | The name of the service endpoint to call. For example, the FAS.Configuration service exposes and endpoint name “ConfigurationService” |
| Domain | The name of the domain in which the service endpoint exists. For example, in our DEV environment, our domain is “dev.tcsc.com” |
| Protocols | The container element for protocol definitions |
| Security | The container element for security definition |

## Security Element

| Entry | Description |
| --- | --- |
| SpnIdentity[[1]](#footnote-1) | The name of the Identity assigned to access the service endpoint. This SPN is usually assigned to an endpoint by network administrator. |

## Protocols Element

| Entry | Description |
| --- | --- |
| Key (Required) | The protocol key as defined in DNS |
| Value (Required) | The protocol value |

The AutoDiscovery.config file should contain a configuration section for the TCSC Logger. Configuration of the TCSC Logger is not covered in this document, but is necessary for troubleshooting if the AutoDiscovery process encounters an error and must write an error message to the log file.

# Testing

The AutoDiscovery solution has been integrated into each application and service in the FAS.Sentinel.Serialization solution. The best method for testing the process is to back up any existing configuration files by copying them to another directory and then launch the application or service with no application configuration file present in the execution directory. If everything is properly configured, a new application configuration file is retrieved from the configuration service and the application launches. If the application encounters a problem during the AutoDiscovery process, the AutoDiscovery API writes an error to the log file.

1. Service Principal Name (SPN) is a unique identifier of a service instance. Refer to “[Service Principal Names](https://msdn.microsoft.com/en-us/library/windows/desktop/ms677949%28v=vs.85%29.aspx?f=255&MSPPError=-2147217396)” at <https://msdn.microsoft.com/en-us/library/windows/desktop/ms677949%28v=vs.85%29.aspx?f=255&MSPPError=-2147217396> for more specific information. [↑](#footnote-ref-1)