Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.

- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights.** All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/19/2010</td>
<td>1.0</td>
<td>Major</td>
<td>Initial Availability</td>
</tr>
<tr>
<td>03/31/2010</td>
<td>1.01</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>04/30/2010</td>
<td>1.02</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/07/2010</td>
<td>1.03</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>06/29/2010</td>
<td>1.04</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>07/23/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>09/27/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>03/18/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>06/10/2011</td>
<td>1.04</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>01/20/2012</td>
<td>1.5</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>04/11/2012</td>
<td>1.5</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>07/16/2012</td>
<td>1.5</td>
<td>No change</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction .............................................................................................................. 6
  1.1 Glossary ............................................................................................................ 6
  1.2 References ........................................................................................................ 6
    1.2.1 Normative References ............................................................................. 7
    1.2.2 Informative References ............................................................................. 7
  1.3 Overview ............................................................................................................ 7
  1.4 Relationship to Protocols and Other Structures ............................................. 8
    1.4.1 User Store Configuration File Format ............................................... 8
    1.4.2 Principal Aliaser Configuration File Format ........................................ 9
  1.5 Applicability Statement .................................................................................... 9
    1.5.1 User Store Configuration File Format ............................................... 9
    1.5.2 Principal Aliaser Configuration File Format ........................................ 10
  1.6 Versioning and Localization ............................................................................ 10
  1.7 Vendor-Extensible Fields .................................................................................. 10

2 Structures ............................................................................................................ 11
  2.1 Common File Format ....................................................................................... 11
    2.1.1 Global Elements ...................................................................................... 11
      2.1.1.1 Configuration .................................................................................... 11
    2.1.2 Complex Types ......................................................................................... 11
      2.1.2.1 CT_ConfigSections ............................................................................ 11
      2.1.2.2 CT_Configuration ............................................................................. 11
      2.1.2.3 CT_Constructor .................................................................................. 12
      2.1.2.4 CT_Container .................................................................................... 12
      2.1.2.5 CT_Containers .................................................................................. 13
      2.1.2.6 CT_Dependency ............................................................................... 13
      2.1.2.7 CT_Instances .................................................................................... 13
      2.1.2.8 CT_Lifetime ....................................................................................... 14
      2.1.2.9 CT_Param .......................................................................................... 14
      2.1.2.10 CT_Property ..................................................................................... 14
      2.1.2.11 CT_Section ...................................................................................... 15
      2.1.2.12 CT_Type .......................................................................................... 16
      2.1.2.13 CT_TypeAlias .................................................................................. 16
        2.1.2.13.1 TypeAlias[1] Element ................................................................. 17
        2.1.2.13.2 TypeAlias[2] Element ................................................................. 17
      2.1.2.14 CT_TypeAliases .............................................................................. 17
      2.1.2.15 CT_TypeConfig .............................................................................. 18
      2.1.2.16 CT_Types ........................................................................................ 18
      2.1.2.17 CT_Unity ........................................................................................ 19
      2.1.2.18 CT_Value ......................................................................................... 19
    2.1.3 Simple Types ............................................................................................... 20
    2.2 User Store Configuration File Format ......................................................... 20
      2.2.1 Type Elements ......................................................................................... 20
        2.2.1.1 Domain Type Element .................................................................... 20
          2.2.1.1.1 Type Element ........................................................................... 20
          2.2.1.1.2 Param Element ......................................................................... 20
          2.2.1.1.3 Property Element ..................................................................... 21
        2.2.1.2 PreFilterGeneratorSimple Type Element ........................................ 22
          2.2.1.2.1 Type Element ........................................................................... 22
          2.2.1.2.2 Param Element ......................................................................... 22
1 Introduction

This document specifies the Search Authorization Configuration File Format used by the search authorization manager service and the search authorization worker component of the query processing service described in [MS-FSO]. The file formats are used to provide secure search capability to ensure that users view only the search results for which they are authorized. The User Store Configuration File Format is used to configure the search authorization worker component for each source of user objects and group objects. The Principal Aliaser Configuration File Format is used to configure the search authorization worker component for mapping user and group objects to other user and group objects.

Sections 1.7 and 2 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

- group object
- security identifier (SID)
- user object
- UTF-8

The following terms are defined in [MS-OFCGLOS]:

- FAST Search Authorization (FSA)
- managed property
- principal aliaser
- principal aliaser identifier
- principal aliasing
- query processing
- security principal
- security principal identifier
- user name
- user security filter
- user store
- user store identifier
- XML principal aliaser
- XML schema
- XML schema definition (XSD)

The following terms are specific to this document:

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as described in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the technical documents, which are updated frequently. References to other documents include a publishing year when one is available.
1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.


1.2.2 Informative References


[MS-FSO] Microsoft Corporation, "FAST Search System Overview".


[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

1.3 Overview

Secure search ensures that only authorized search results are returned. This occurs in two phases. In the first phase, the customer's content repositories are traversed to create indexes. Authorization managed properties that identify the user objects and group objects which are granted or denied access are added to the indexes for each item.

In the second phase, a query comes from the user and the indexes quickly identify search results. In this phase, secure search consists of authenticating the user and rewriting the query so that the indexes return only authorized search results. The query processing service described in [MS-FSO] rewrites the query by intersecting the original query with the user security filter. The user security filter uses the authorization managed properties to limit the query results to items for which the user is authorized. It is generated by the FAST Search Authorization (FSA) worker component which is part of the query processing service.

The FSA worker component uses the groups with which the user object is associated to generate the user security filter. A user store is responsible for providing group membership information for its user objects. This document describes the user store configuration files that enable the FSA worker
component to interact with the various kinds of user stores. The user store contains the groups that a user object belongs to, which enables the FSA worker component to create the user security filter.

Some users have identities in multiple user stores. For example, a claims user can have a corresponding account in a collaborative business application. To generate the user security filter, the FSA worker component requires the identities and group objects that are associated with the user object in all user stores. The security principal identifier is not necessarily the same in all user stores. The FSA worker component uses principal aliasing to map user objects and group objects from one user store into another.

Principal aliasers map user objects and group objects to other user stores. This document describes the principal aliaser configuration files that enable the FSA worker component to interact with the various kinds of principal aliasers. The principal aliasers describe the other user stores that are associated with a user object or group object, and this enables the FSA worker component to create the user security filter.

The user store and principal aliaser configuration file formats are XML files that share a common schema. There is one user store configuration file for each user store, and one principal aliaser configuration file for each principal aliaser. Much of the file content is identical. The parts that are the same are described in section 2.1. Section 2.2 describes the elements that are unique for the user store configuration file, and section 2.3 describes the elements that are unique for the principal aliaser configuration file. If an XML element or attribute is not described in section 2.2 or 2.3, respectively, then the protocol uses section 2.1.

The difference between the two file formats are in the type elements. Section 2.2 describes the types of the type element in a user store configuration file. Section 2.3 describes the type elements in a principal aliaser configuration file. There are specific requirements for each type of type element. If a child element or attribute for a type element is not described in section 2.2 or 2.3, then the protocol uses section 2.1.

1.4 Relationship to Protocols and Other Structures

The user store configuration file format is not dependent on any other structure, nor is the principal aliaser configuration file format. Both file formats are used as payload in the Search Authorization Synchronization Protocol TransferFile message, as described in [MS-FSSAS] section 2.2.2.1. The FSA Managers Service is described in [MS-FSO]. The FSA worker component is part of the query processing service described in [MS-FSO].

1.4.1 User Store Configuration File Format

The following figure describes the relationship between the user store configuration file and the FSA components.
1.4.2 Principal Aliaser Configuration File Format

The following figure describes the relationship between the principal aliaser configuration file and the FSA components.

![Diagram of Principal Aliaser Configuration File Relationships]

1.5 Applicability Statement

The file format specified in section 2.1 was designed to configure modules on the FSA worker component of the query processing service [MS-FSO]. This document describes the two uses of that format, the user store (section 2.2) and principal aliaser (section 2.3) configuration file formats. Implementation-dependent methods are used to protect them at all times because they contain information relevant to application security.

1.5.1 User Store Configuration File Format

The user store configuration file format is applicable to accessing, modifying, or creating the configuration of a user store on the FSA worker component. A user store provides user and group information to the FSA worker component that is used to construct a user security filter.
1.5.2 Principal Aliaser Configuration File Format

The principal aliaser configuration file format is applicable to accessing, modifying, or creating the configuration of a principal aliaser on the FSA worker component of the query processing service. A principal aliaser specifies mappings of security identifiers (SIDs) between different user stores for the FSA worker component. The mappings are used to construct a user security filter.

1.6 Versioning and Localization

None.

1.7 Vendor-Extensible Fields

None.
2  Structures

Throughout this document, the xs namespace is http://www.w3.org/2001/XMLSchema. XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2.

2.1  Common File Format

The user store (section 2.2) and principal aliaser (section 2.3) configuration files are valid standalone XML files that MUST use UTF-8 encoding and the same schema.

The user store configuration and principal aliaser configuration files specify how to construct objects on the FSA worker component which is part of the query processing service [MS-FSO]. These objects are the configuration of the FSA worker component.

For more information about the full XML schema, see section 5.

2.1.1  Global Elements

2.1.1.1  Configuration

A CT_Configuration element MUST be the root element of the user store configuration and principal aliaser configuration files. The following XML schema definition (XSD), as specified in [XMLSCHEMA1] section 2.1, fragment specifies the contents of this element.

```xml
<xsd:element name="configuration" type="CT_Configuration" />
```

2.1.2  Complex Types

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2.

2.1.2.1  CT_ConfigSections

Referenced by: CT_Configuration, configSections

This complex type specifies the configSections element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_ConfigSections">
  <xs:sequence>
    <xs:element name="section" type="CT_Section" />
  </xs:sequence>
</xs:complexType>
```

Child Elements:

section: A CT_Section element that specifies the element that contains the configuration and the type that processes it.

2.1.2.2  CT_Configuration

Referenced by: configuration
A complex type that specifies the configuration element. The configuration element is the root element of the user store configuration and principal allaiser configuration files. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Configuration">
  <xs:sequence>
    <xs:element name="configSections" type="CT_ConfigSections" />
    <xs:element name="unity" type="CT_Unity" />
  </xs:sequence>
</xs:complexType>
```

**Child Elements:**

**configSections:** A **CT_ConfigSections** element that specifies elements that will follow it. The configSections element MUST contain exactly one section element. The section element specifies that a unity element follows the configSections element.

**unity:** A **CT_Unity** element that specifies the configuration for a user store configuration or principal allaiser configuration file.

### 2.1.2.3 CT_Constructor

**Referenced by:** **CT_TypeConfig**, constructor

A complex type that specifies the constructor element. It is used to construct an object. It and its child elements MUST NOT be added unless they are contained the type element in specified in section 2.2 or 2.3. An type element in section 2.2 or 2.3 specifies the type/typeConfig/constructor/param elements. The order of the param elements MUST NOT be changed. The following fragment XSD ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Constructor">
  <xs:sequence>
    <xs:element minOccurs="unbounded" name="param" type="CT_Param" />
  </xs:sequence>
</xs:complexType>
```

**Child Elements:**

**param:** A **CT_Param** element that specifies the parameter used to construct the object.

### 2.1.2.4 CT_Container

**Referenced by:** **CT_Containers**, container

A complex type that specifies the container element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Container">
  <xs:sequence>
    <xs:element name="types" type="CT_Types" />
    <xs:element name="instances" type="CT_Instances" />
  </xs:sequence>
</xs:complexType>
```
Child Elements:

types: A CT_Types element that specifies the type elements that specify the construction of the configuration objects.

instances: A CT_Instances element that MUST exist.

2.1.2.5 CT_Containers

Referenced by: CT_Unity, containers

A complex type that specifies the containers element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Containers">
    <xs:sequence>
        <xs:element name="container" type="CT_Container" />  
    </xs:sequence>
</xs:complexType>
```

Child Elements:

container: A CT_Container element.

2.1.2.6 CT_Dependency

Referenced by: CT_Param, CT_Property, CT_TypeConfig, dependency

A complex type that specifies the dependency element. This element specifies the complex type that is a child of a param or property element. The CT_Dependency element specifies that the value of a typeConfig, param, or property element is specified in a type element. A type element in section 2.2 or 2.3 specifies the required elements that are referenced by the name attribute of the dependency element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Dependency">
    <xs:attribute name="name" type="xs:string" use="optional" />  
</xs:complexType>
```

Attributes:

name: An xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies the name attribute of a type element.

2.1.2.7 CT_Instances

Referenced by: CT_Container, instances

A complex type that specifies an instances element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Instances">
</xs:complexType>
```
2.1.2.8  CT_Lifetime

*Referenced by: CT_Type, lifetime*

A complex type that specifies a lifetime element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Lifetime">
  <xsd:attribute name="type" type="xsd:string" use="required" fixed="singleton" />
</xsd:complexType>
```

**Attributes:**

- **type:** An *xs:string* attribute ([XMLSCHEMA2] section 3.2.1) that MUST be "singleton".

2.1.2.9  CT_Param

*Referenced by: CT_Constructor, param*

A complex type that specifies a param element for constructing an object. A type element in section 2.2 or 2.3 specifies the type/typeConfig/constructor/param elements. The order of the param elements MUST NOT be changed. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xsd:complexType name="CT_Param">
  <xsd:choice>
    <xsd:element minOccurs="0" name="value" type="CT_Value" />
    <xsd:element minOccurs="0" name="dependency" type="CT_Dependency" />
  </xsd:choice>
  <xsd:attribute name="name" type="xs:string" use="required" />
  <xsd:attribute name="parameterType" type="xs:string" use="required" />
</xsd:complexType>
```

**Child Elements:**

- **value:** A CT_Value element that specifies the value of the constructor parameter.
- **dependency:** A CT_Dependency element that specifies that a constructor parameter is another type element.

**Attributes:**

- **name:** An *xs:string* attribute ([XMLSCHEMA2] section 3.2.1) that specifies the name of the constructor parameter.
- **parameterType:** An *xs:string* attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the constructor parameter. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and line feed.

2.1.2.10  CT_Property

*Referenced by: CT_TypeConfig, property*

[XMLSCHEMA1] — v20120630
Search Authorization Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
A complex type that specifies a **property** element. The **type** element in section 2.2 or 2.3 specifies the type/typeConfig/property elements. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Property">
  <xs:sequence>
    <xs:element minOccurs="0" name="value" type="CT_Value" />
    <xs:element minOccurs="0" name="dependency" type="CT_Dependency" />
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required" />
  <xs:attribute name="propertyType" type="xs:string" use="required" />
</xs:complexType>
```

**Child Elements:**

**value:** A **CT_Value** element that specifies the value of the **property** element.

**dependency:** A **CT_Dependency** element that specifies that the **property** value is another **type** element.

**Attributes:**

**name:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that specifies the name of the **property** element.

**propertyType:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the **property** value. This attribute MUST ignore leading and trailing white space, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and line feed.

### 2.1.2.11 CT_Section

**Referenced by:** CT_ConfigSections, section

A complex type that specifies the **section** element. The **section** element specifies the element that contains the configuration and the type that processes it. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Section">
  <xs:attribute name="name" type="xs:string" use="required" fixed="unity" />
  <xs:attribute name="type" type="xs:string" use="required" />
</xs:complexType>
```

**Attributes:**

**name:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that MUST be "unity". The **name** attribute specifies the element that contains the configuration.

**type:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that MUST be "Microsoft.Practices.Unity.Configuration.UnityConfigurationSection, Microsoft.Practices.Unity.Configuration, Version=1.2.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c". This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed. The **type** attribute specifies the type that will process the **configuration** element specified by the **name** attribute.
2.1.2.12 CT_Type

Referenced by: CT_Types, type

A complex type that specifies a type element representing a configuration object. This element and its child elements contain configuration information specific to a user store or principal aliaser configuration file. Each configuration file contains multiple type elements. The type element in section 2.2 or 2.3 specifies the child elements and attribute values. The following XSD fragment specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Type">
  <xs:sequence>
    <xs:element name="lifetime" type="CT_Lifetime" />
    <xs:element name="typeConfig" type="CT_TypeConfig" />
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required" />
  <xs:attribute name="name" type="xs:string" use="optional" />
  <xs:attribute name="mapTo" type="xs:string" use="optional" />
</xs:complexType>
```

Child Elements:

**lifetime:** A CT_Lifetime element that specifies the lifetime of the configuration object.

**typeConfig:** A CT_TypeConfig element that specifies creation of the configuration object.

Attributes:

**type:** A required xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the configuration object. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed. The type value is specified in sections 2.2 and 2.3.

**name:** An optional xs:string attribute that specifies the name of this type element. When required for a type element, name attributes are specified in sections 2.2 and 2.3.

**mapTo:** An optional xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies a related type of the configuration object. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed. When required for a type element, the mapTo attributes are specified in sections 2.2 and 2.3.

2.1.2.13 CT_TypeAlias

Referenced by: CT_TypeAliases, typeAlias

A complex type that specifies a typeAlias element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_TypeAlias">
  <xs:attribute name="alias" type="xs:string" use="required" />
  <xs:attribute name="type" type="xs:string" use="required" />
</xs:complexType>
```

Attributes:
**alias**: An *xs:string* attribute ([XMLSCHEMA2] section 3.2.1) that specifies an alternate name to use for a type.

**type**: An *xs:string* attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the *type alias*. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.

### 2.1.2.13.1 TypeAlias[1] Element

The first *typeAlias* element MUST contain the attributes specified in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>MUST be &quot;singleton&quot;.</td>
</tr>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.Practices.Unity.ContainerControlledLifetimeManager,Microsoft.Practices.Unity, version=1.2.0.0, PublicKeyToken=71e9bce111e9429c&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.</td>
</tr>
</tbody>
</table>

### 2.1.2.13.2 TypeAlias[2] Element

The second *typeAlias* element MUST contain the attributes specified in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>MUST be &quot;ILog&quot;.</td>
</tr>
</tbody>
</table>

### 2.1.2.14 CT_TypeAliases

**Referenced by**: CT_Unity, typeAliases

A complex *type* that specifies a *typeAliases* element. The following XSD attribute ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_TypeAliases">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="typeAlias" type="CT_TypeAlias" />
  </xs:sequence>
</xs:complexType>
```

**Child Elements**:

**typeAlias**: A CT_TypeAlias element.
2.1.2.15  CT_TypeConfig

Referenced by: CT_Type, typeConfig

A complex type that specifies a typeConfig element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_TypeConfig">
  <xs:choice>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="dependency" type="CT_Dependency" />
    <xs:sequence>
      <xs:element minOccurs="0" name="constructor" type="CT_Constructor" />
      <xs:element minOccurs="0" maxOccurs="unbounded" name="property" type="CT_Property" />
    </xs:sequence>
  </xs:choice>
  <xs:attribute name="extensionType" type="xs:string" use="required" />
  <xs:attribute name="type" type="xs:string" use="optional" />
  <xs:attribute name="validateFunc" type="xs:string" use="optional" />
</xs:complexType>
```

Child Elements:

dependency: A CT_Dependency element that specifies dependency elements that are contained by this type element.

constructor: A CT_Constructor element that specifies construction of the type element.

property: A CT_Property element that specifies properties of the type element.

value: A CT_Value element that specifies the value of the type element.

Attributes:

extensionType: An xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies an object to convert the type element into a configuration object. This attribute MUST be "Microsoft.Practices.Unity.Configuration.TypeInjectionElement,Microsoft.Practices.Unity.Configuration, version=1.2.0.0, PublicKeyToken=71e9bce111e9429c" unless specified otherwise for a particular typeConfig element in sections 2.2 or 2.3. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.

type: An xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the configuration object. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed. The value of this attribute will be specified in sections 2.2 or 2.3.

validateFunc: An xs:string attribute ([XMLSCHEMA2] section 3.2.1) that specifies an object the validates the configuration object. This attribute MUST be omitted unless specified otherwise for a particular typeConfig element in sections 2.2 or 2.3.

2.1.2.16  CT_Types

Referenced by: CT_Container, types
A complex type that specifies a **types** element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Types">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="type" type="CT_Type" />
  </xs:sequence>
</xs:complexType>
```

**Child Elements:**

**type:** A **CT_Type** element that specifies the construction of a **configuration** object.

### 2.1.2.17 CT_Unity

**Referenced by:** **CT_Configuration**, unity

A complex type that specifies a **unity** element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type.

```xml
<xs:complexType name="CT_Unity">
  <xs:sequence>
    <xs:element name="typeAliases" type="CT_TypeAliases" />
    <xs:element name="containers" type="CT_Containers" />
  </xs:sequence>
</xs:complexType>
```

**Child Elements:**

**typeAliases:** A **CT_TypeAliases** element.

**containers:** A **CT_Containers** element.

### 2.1.2.18 CT_Value

**Referenced by:** **CT_Param**, **CT_Property**, **CT_TypeConfig**, value

A complex type that specifies a **value** element which is the value of a **constructor** or **property** element. The following XSD fragment ([XMLSCHEMA1] section 2.1) specifies the contents of this complex type. The attribute values are specified for each **type** element in sections 2.2 and 2.3.

```xml
<xs:complexType name="CT_Value">
  <xs:attribute name="value" type="xs:string" use="optional" />
  <xs:attribute name="type" type="xs:string" use="optional" />
  <xs:attribute name="typeConverter" type="xs:string" use="optional" />
</xs:complexType>
```

**Attributes:**

**value:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that specifies the literal value.

**type:** An **xs:string** attribute ([XMLSCHEMA2] section 3.2.1) that specifies the type of the value. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal
sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.

**typeConverter:** An `xs:string` attribute ([XMLSCHEMA2] section 3.2.1) that specifies the object used to convert the value into the required type. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed. This attribute cannot be changed and MUST NOT be used unless it is specified as part of a type element in sections 2.2 and 2.3.

### 2.1.3 Simple Types

None.

### 2.2 User Store Configuration File Format

This section specifies how the elements and attributes specified in section 2.1 are modified and combined to create a user store configuration file.

#### 2.2.1 Type Elements

This section specifies the types of type elements and associated child elements that are in a user store configuration file. Section 2.2.2 specifies how to combine them into a user store configuration file.

##### 2.2.1.1 Domain Type Element

This section specifies the differences between the domain type element and the type elements and child elements specified in section 2.1. All user store configuration files MUST contain a domain type element.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

##### 2.2.1.1.1 Type Element

The type element MUST contain the attribute that is specified in the following table.

<table>
<thead>
<tr>
<th>Attribute Name</th>
<th>Description</th>
</tr>
</thead>
</table>

##### 2.2.1.2 Param Element

The following table specifies the param child elements of the type/typeConfig/constructor element for a domain type element. All name and parametertype attributes associated with the param element MUST contain the specified values in the specified order. Each child element MUST contain the attributes that are specified in the following table.
<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>logger</td>
<td>ILog</td>
<td>MUST be a <code>dependency</code> element that contains no attributes.</td>
</tr>
<tr>
<td>id</td>
<td>System.String</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute that specifies the <code>user store identifier</code> for the user store configuration file.</td>
</tr>
<tr>
<td>description</td>
<td>System.String</td>
<td>MUST be either a <code>value</code> element that contains no attributes, or a <code>value</code> element that contains a <code>value</code> attribute that specifies a short description of the user store.</td>
</tr>
<tr>
<td>contentType</td>
<td>System.String</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute. If the user store configuration file is for a claims user store, the <code>value</code> attribute MUST be &quot;Claims&quot; as specified in section 2.2.2.1. If the user store configuration file is for a local cache user store, the <code>value</code> attribute MUST be &quot;LotusNotes&quot; as specified in section 2.2.2.2.</td>
</tr>
<tr>
<td>preFilterGenerator</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator, Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.</td>
<td>MUST be a <code>dependency</code> element that contains a <code>name</code> attribute. If the user store configuration file is for a claims user store, the <code>name</code> attribute MUST be &quot;SimplePreFilterGenerator&quot; (section 2.2.2.1). If the user store configuration file is for a local cache user store, the <code>name</code> attribute MUST be &quot;LotusNotesPreFilterGenerator&quot; (section 2.2.2.2).</td>
</tr>
</tbody>
</table>

### 2.2.1.1.3 Property Element

The following table specifies the `property` child element of the `type/typeConfig` elements. All `property name` and `propertyType` attributes MUST be the values specified in the following table.
### Table 1: Value of name attribute, Value of propertyType attribute, and Child element

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore</td>
</tr>
<tr>
<td></td>
<td>leading and trailing whitespace, and it MUST also collapse internal</td>
</tr>
<tr>
<td></td>
<td>sequences of whitespace into a single space. White space characters are</td>
</tr>
<tr>
<td></td>
<td>space, tab, carriage return, and line feed.</td>
</tr>
<tr>
<td></td>
<td>MUST be a dependency element that contains a name attribute. If the user</td>
</tr>
<tr>
<td></td>
<td>store configuration file is for a claims user store, the name attribute</td>
</tr>
<tr>
<td></td>
<td>MUST be &quot;ClaimsUserMonitor&quot; (section 2.2.2.1). If the user store configuration</td>
</tr>
<tr>
<td></td>
<td>file is for a local cache user store, the name attribute MUST be</td>
</tr>
<tr>
<td></td>
<td>&quot;LocalCacheUserMonitor&quot; (section 2.2.2.2).</td>
</tr>
</tbody>
</table>

### 2.2.1.2 PreFilterGeneratorSimple Type Element

The PreFilterGeneratorSimple type element MUST be only in user store configuration files for claims user stores. This section specifies only the differences from the type elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

#### 2.2.1.2.1 Type Element

The type element MUST have the following attributes with the specified values.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MUST be &quot;SimplePreFilterGenerator&quot;.</td>
</tr>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator,</td>
</tr>
<tr>
<td></td>
<td>Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore</td>
</tr>
<tr>
<td></td>
<td>leading and trailing whitespace, and it MUST also collapse internal</td>
</tr>
<tr>
<td></td>
<td>sequences of whitespace into a single space. White space characters are</td>
</tr>
<tr>
<td></td>
<td>space, tab, carriage return, and line feed.</td>
</tr>
</tbody>
</table>

#### 2.2.1.2.2 Param Element

The following table specifies the param child element of the type/typeConfig/constructor elements for a PreFilterGeneratorSimple type element. All name and parameterType attributes associated
with the `param` element MUST contain the specified value. Each child element MUST contain the attributes that are specified in the following table.

<table>
<thead>
<tr>
<th>name attribute</th>
<th>parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>logger</td>
<td>ILog</td>
<td>MUST be a dependency element that contains no attributes.</td>
</tr>
</tbody>
</table>

### 2.2.1.2.3 Property Element

The following table specifies the `property` child elements of the `type/typeConfig` elements. All `name` and `propertyType` attributes of the `typeConfig` element MUST contain the values specified in the following table. All child elements are `value` elements that contain `value` attributes whose value is specified in the following table.

<table>
<thead>
<tr>
<th>Value of name attribute</th>
<th>Value of propertyType attribute</th>
<th>Child element value attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowPrefix</td>
<td>System.String</td>
<td>The <code>value</code> attribute MUST be &quot;docacl:&quot;.</td>
</tr>
<tr>
<td>DenyPrefix</td>
<td>System.String</td>
<td>The <code>value</code> attribute MUST be &quot;docacl:9&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterPattern</td>
<td>System.String</td>
<td>If the user store configuration file is for a claims user store, the <code>value</code> attribute MUST be &quot;[^a-z0-9]&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterHandling</td>
<td>System.String</td>
<td>The <code>value</code> attribute MUST be &quot;encode&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterEncodingPrefix&quot;</td>
<td>System.String</td>
<td>The <code>value</code> attribute MUST be &quot;3nc0dd&quot;.</td>
</tr>
</tbody>
</table>

### 2.2.1.3 LotusNotesPreFilterGenerator Type Element

This `type` element MUST be only in user store configuration files for local cache user stores. This section specifies only the differences between the elements and the `type` elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

#### 2.2.1.3.1 Type Element

The `type` element MUST contain the attributes specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MUST be &quot;LotuseNotesPreFilterGenerator&quot;.</td>
</tr>
</tbody>
</table>
2.2.1.3.2 Param Element

This specifies the param child elements of the type/typeConfig/constructor elements for a LotusNotesPreFilterGenerator type element as described in [LotusNotes]. All name and parameterType attributes associated with the param element MUST contain the specified values in the specified order. Each child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>logger</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>useDenyOnDefault</td>
<td>System.Boolean</td>
<td>MUST be a value element with value and type attributes. The type attribute MUST be &quot;System.Boolean&quot;. The value attribute MUST contain the value true if the user security filter denies user object access by default, false otherwise.</td>
</tr>
<tr>
<td>useDocumentSecurity</td>
<td>System.Boolean</td>
<td>MUST be a value element with value and type attributes. The type attribute MUST be &quot;System.Boolean&quot;. The value attribute MUST contain the value true if the collaborative business applications protocol server was configured to use &quot;document-level&quot; security, as described in [LotusNotes]. Otherwise, it MUST contain the value false.</td>
</tr>
<tr>
<td>useWildcardSecurity</td>
<td>System.Boolean</td>
<td>MUST be a value element with value and type attributes. The type attribute MUST be &quot;System.Boolean&quot;. The value attribute MUST be true if the collaborative business applications protocol server was configured to use &quot;wild card&quot; security, as described in [LotusNotes]. Otherwise, it MUST contain the value false.</td>
</tr>
<tr>
<td>useViewSecurity</td>
<td>System.Boolean</td>
<td>MUST be a value element with value and type attributes. The type attribute MUST be &quot;System.Boolean&quot;. The value attribute MUST be true if the collaborative business applications protocol server was configured to use &quot;view&quot; security, as described in [LotusNotes].</td>
</tr>
<tr>
<td>Value of param name attribute</td>
<td>Value of param parameterType attribute</td>
<td>Child element value attribute</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>AllowPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;docacl:&quot;.</td>
</tr>
<tr>
<td>DenyPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;docacl:9&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterPattern</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;.*&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterHandling</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;encodeWithPrefix&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterEncodingPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;3nc0dd&quot;.</td>
</tr>
</tbody>
</table>

### 2.2.1.3.3 Property Element

This specifies the `property` child element of the `type/typeConfig` elements. All `typeConfig` element `name` and `propertyType` attributes MUST be as specified. All child elements MUST be `value` elements that contain the `value` attribute specified in the following table.

<table>
<thead>
<tr>
<th>Value of name attribute</th>
<th>Value of propertyType attribute</th>
<th>Child element value attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>AllowPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;docacl:&quot;.</td>
</tr>
<tr>
<td>DenyPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;docacl:9&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterPattern</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;.*&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterHandling</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;encodeWithPrefix&quot;.</td>
</tr>
<tr>
<td>NonIndexableCharacterEncodingPrefix</td>
<td>System.String</td>
<td>The value attribute MUST be &quot;3nc0dd&quot;.</td>
</tr>
</tbody>
</table>

### 2.2.1.4 LocalCacheUserMonitor Type Element

The `LocalCacheUserMonitor` `type` element MUST be only in user store configuration files for local cache user stores. This section specifies only the differences between these elements and the `type` elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

### 2.2.1.4.1 Type Element

The `type` element MUST contain the attributes that are specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MUST be &quot;LocalCacheUserMonitor&quot;.</td>
</tr>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IUserMonitor, Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.</td>
</tr>
</tbody>
</table>
2.2.1.4.2 TypeConfig Element

The `typeConfig` element MUST contain the attributes that are specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validateFunc</td>
<td>MUST be &quot;Validate&quot;.</td>
</tr>
</tbody>
</table>

2.2.1.4.3 Param Element

The following table specifies the `param` child elements of the `type/typeConfig/constructor` elements for a LocalCacheUserMonitor `type` element. All `name` and `parametertype` attributes associated with the `param` element MUST contain the specified values in the specified order. Each child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>largeFileStorage</td>
<td>System.String</td>
<td>MUST be a dependency element whose <code>name</code> attribute contains &quot;LargeStorePath&quot;.</td>
</tr>
<tr>
<td>databaseFileName</td>
<td>System.String</td>
<td>MUST be a value element whose <code>value</code> attribute contains the name of the local cache user store file name as specified in [MS-FSSADFF] section 2.2.</td>
</tr>
</tbody>
</table>

2.2.1.4.4 Property Element

This specifies the `property` child elements of the `type` and the `typeConfig` elements. All `name` and `propertyType` attributes associated with the `typeConfig` element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of name attribute</th>
<th>Value of propertyType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>System.String</td>
<td>MUST be a value element whose <code>value</code> attribute contains &quot;LocalCacheUserMonitor&quot;.</td>
</tr>
<tr>
<td>Value of name attribute</td>
<td>Value of propertyType attribute</td>
<td>Child element</td>
</tr>
<tr>
<td>-------------------------</td>
<td>---------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>InitialCapacity</td>
<td>System.Int32</td>
<td>MUST be a value element with value and type attributes. The type attribute contains &quot;System.Int32&quot;. The value attribute specifies the number of entity records in the initial fixed-length section of the database file, as specified for the InitialCapacity field in [MS-FSSADFF] section 2.2.2.1.</td>
</tr>
<tr>
<td>IDSize</td>
<td>System.Int32</td>
<td>MUST be a value element with value and type attributes. The type attribute contains &quot;System.Int32&quot;. The value attribute specifies the number of bytes to use to store a security principal identifier in the database file, as specified for the IDLength field in [MS-FSSADFF] section 2.2.2.1.</td>
</tr>
<tr>
<td>NameSize</td>
<td>System.Int32</td>
<td>MUST be a value element with value and type attributes. The type attribute contains &quot;System.Int32&quot;. The value attribute specifies the number of bytes to use to store a security principal alias in the database file, as specified for the NameLength field in [MS-FSSADFF] section 2.2.2.1.</td>
</tr>
<tr>
<td>MaxParents</td>
<td>System.Int32</td>
<td>MUST be a value element with value and type attributes. The type attribute contains &quot;System.Int32&quot;. The value attribute specifies the maximum number of nested parent security principals for a specific security principal, as specified for the ParentCount field in [MS-FSSADFF] section 2.2.2.1.</td>
</tr>
<tr>
<td>CaseSensitiveLookup</td>
<td>System.Boolean</td>
<td>MUST be a value element with value and type attributes. The type attribute contains &quot;System.Boolean&quot;. The value attribute MUST be &quot;true&quot; if security principal identifiers are case sensitive, &quot;false&quot; otherwise, as specified for the CaseSensitiveLookup field in [MS-FSSADFF] section 2.2.2.1.</td>
</tr>
</tbody>
</table>

The following table specifies fields in the local cache user store file header ([MS-FSSADFF] section 2.2.2.1) that MUST have identical values in the corresponding property elements of a LocalCacheUserMonitor type element.

<table>
<thead>
<tr>
<th>Header field in Local Cache User Store file</th>
<th>property element name attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>InitialCapacity</td>
<td>InitialCapacity</td>
</tr>
<tr>
<td>IDLength</td>
<td>IDSize</td>
</tr>
<tr>
<td>NameLength</td>
<td>NameSize</td>
</tr>
<tr>
<td>ParentCount</td>
<td>MaxParents</td>
</tr>
<tr>
<td>CaseSensitiveLookup</td>
<td>CaseSensitiveLookup</td>
</tr>
</tbody>
</table>
2.2.1.5 ClaimsUserMonitor Type Element

The ClaimsUserMonitor type element MUST be only in user store configuration files for claims user stores. This section specifies only the differences between these elements and the type elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

2.2.1.5.1 Type Element

The type element MUST contain the attributes that are specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MUST be &quot;ClaimsUserMonitor&quot;.</td>
</tr>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IUserMonitor, Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and line feed.</td>
</tr>
</tbody>
</table>

2.2.1.5.2 Param Element

This specifies the param child elements of the type/typeConfig/constructor elements for a ClaimsUserMonitor type element. All name and parameterType attributes associated with the param element MUST contain the specified values in the specified order. Each child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of name attribute</th>
<th>Value of parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>Value of name attribute</td>
<td>Value of parameterType attribute</td>
<td>Child element</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td></td>
<td>characters are space, tab, carriage return, and line feed.</td>
<td></td>
</tr>
<tr>
<td>issuer</td>
<td>System.String</td>
<td>MUST contain the value &quot;@ALL@&quot; or the value that specifies the original issuer of the claim.</td>
</tr>
</tbody>
</table>

2.2.2 Standard User Store

There are two types of standard user store, the claims user store and the local cache user store. Differences between them are specified in the following sections.

2.2.2.1 Claims User Store

A claims user store configuration file MUST contain the following type elements:

- Domain type element (section 2.2.1.1).
- PreFilterGeneratorSimple type element (section 2.2.1.2).
- ClaimsUserMonitor type element (section 2.2.1.5).

2.2.2.2 Local Cache User Store

A local cache user store configuration file MUST contain the following type elements:

- Domain type element (section 2.2.1.1).
- LotusNotesPreFilterGenerator type element (section 2.2.1.3).
- LocalCacheUserMonitor type element (section 2.2.1.4).

2.3 Principal Aliaser Configuration File Format

This section specifies how the elements and attributes specified in section 2.1 are modified and combined to create a principal aliaser configuration file.

2.3.1 Type Elements

This section specifies the various kinds of type instances and child elements that are in a principal aliaser configuration file. Section 2.3.2 specifies how these instances are combined into a principal aliaser configuration file.
2.3.1.1 XMLPrincipalAliaser Type Element

The XMLPrincipalAliaser type element MUST be contained only in configuration files for XML principal aliasers. This section specifies only the differences between these elements and the type elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

2.3.1.1.1 Type Element

The type element MUST contain the attributes that are specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IPrincipalAliaser, Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and linefeed.</td>
</tr>
</tbody>
</table>

2.3.1.1.2 Param Element

This specifies the param child elements of the type/typeConfig/constructor elements for an XMLPrincipalAliaser type element. All name and parametertype attributes associated with the param element MUST contain the specified values in the specified order. Each child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameter Type attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>logger</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>id</td>
<td>System.String</td>
<td>MUST be a value element that contains a value attribute that specifies the principal aliaser identifier for the principal aliaser configuration file.</td>
</tr>
<tr>
<td>inputUserStoreId</td>
<td>System.String</td>
<td>MUST be a value element that contains a value attribute that specifies the input user store identifier. The input user store identifier is the user store identifier of the user store that is input to the mapping. User objects in the input user store to user objects in one of the output user stores.</td>
</tr>
<tr>
<td>outputUserStoreIds</td>
<td>System.String</td>
<td>MUST be a value element that contains value, type and typeConverter attributes. The value attribute is a comma separated list of output user store identifiers.</td>
</tr>
<tr>
<td>Value of param name attribute</td>
<td>Value of param parameter Type attribute</td>
<td>Child element</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>xmlFileName</td>
<td>System.String</td>
<td>This <strong>param</strong> element MUST have no child elements if no file with security identifier (SID) mappings is available. If a file is available, this <strong>param</strong> element MUST contain one <strong>value</strong> element that contains a <strong>value</strong> attribute. The <strong>value</strong> attribute MUST be the name of the file that contains the SID mappings. This file format is specified in [MS-FSSADFF] section 2.3.</td>
</tr>
<tr>
<td>dirPath</td>
<td>System.String</td>
<td>MUST be a <strong>dependency</strong> element that contains a <strong>name</strong> attribute whose value is &quot;EXECUTABLE_PATH_STR&quot;.</td>
</tr>
</tbody>
</table>

### 2.3.1.1.3 Property Element

This specifies the **property** child element of the **typeConfig** element. All **typeConfig name** and **propertyType** attributes MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of property name attribute</th>
<th>Value of property propertyType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>InputProperty</td>
<td>System.String</td>
<td>MUST be a <strong>value</strong> element that contains a <strong>value</strong> attribute that specifies the name of the <strong>property</strong> attribute of a security principal that is the input to the mapping. To map a security principal (user or group), the protocol uses the <strong>property</strong> attribute of the input security principal to select mappings in the XML principal aliaser mapping file ([MS-FSSADFF] section 2.3).</td>
</tr>
</tbody>
</table>

### 2.3.1.2 RegExPrincipalAliaser Type Element

The **RegExPrincipalAliaser** element MUST be only in configuration files for regular expression principal aliasers. This section specifies only the differences between these elements and the **type** elements and child elements specified in section 2.1.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity.Contains/containers/container".
2.3.1.2.1 Type Element

The type element MUST contain attributes as specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>MUST be &quot;Microsoft.SharePoint.Search.Extended.Security.IPrincipalAliaser, Microsoft.SharePoint.Search.Extended.Security&quot;. This attribute MUST ignore leading and trailing whitespace, and it MUST also collapse internal sequences of whitespace into a single space. Whitespace characters are space, tab, carriage return, and line feed.</td>
</tr>
</tbody>
</table>

2.3.1.2.2 Param Element

This specifies the param child elements of the type/typeConfig/constructor elements for a RegExPrincipalAliaser type element. All name and parameterType attributes associated with the param element MUST contain the specified values in the specified order. Each param child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>logger</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>id</td>
<td>System.String</td>
<td>MUST be a value element whose value attribute specifies the principal aliaser identifier for the principal aliaser configuration file.</td>
</tr>
<tr>
<td>inputUserStoreId</td>
<td>System.String</td>
<td>MUST be a value element that contains a value attribute that specifies the input user store identifier, which is the identifier of the user store that is input to the mapping. Users in the input user store are mapped to users in one of the output user stores.</td>
</tr>
</tbody>
</table>
## 2.3.1.2.3 Property Element

This specifies the `property` child elements of the `type/typeConfig/type` or `typeConfig` element. All `typeConfig` element `name` and `propertyType` attributes MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of property name attribute</th>
<th>Value of property propertyType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>CaseSensitive</td>
<td>System.Boolean</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute and a <code>type</code> attribute. The <code>value</code> attribute MUST contain <code>true</code> if pattern matching is case sensitive; <code>false</code> otherwise. The <code>type</code> attribute MUST be &quot;System.Boolean&quot;.</td>
</tr>
<tr>
<td>UnicodeCaseSensitive</td>
<td>System.Boolean</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute and a <code>type</code> attribute. The <code>value</code> attribute MUST be <code>true</code> if pattern matching is based on unicode case sensitive matching; <code>false</code> otherwise. The <code>type</code> attribute MUST be &quot;System.Boolean&quot;.</td>
</tr>
</tbody>
</table>
2.3.1.3  RegExMap type Element

The RegExMap type element MUST be only in configuration files for regular expression principal aliases. This section specifies only the differences between these elements and the type elements and child elements specified in section 2.1.

The RegExMap type element specifies a list of RegExMapItem type elements using dependency elements. Each RegExMapItem type element specifies a different mapping of security principals using regular expressions.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

2.3.1.3.1  Type Element

The type element MUST contain the attributes as specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>MUST be &quot;RegExMap&quot;.</td>
</tr>
</tbody>
</table>

2.3.1.3.2  TypeConfig Element

The typeConfig element MUST contain the attributes as specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Attribute value</th>
</tr>
</thead>
</table>
2.3.1.3.3 Dependency Element

The RegExMap type element MUST be associated with child dependency elements. The name attribute of each dependency element MUST be the name attribute of a RegExMapItem type element (section 2.3.1.4) in this file. Each RegExMapItem type element represents a mapping of security principals using the regular expressions in the RegExMapItem type element.

2.3.1.4 RegExMapItem Type Element

The RegExMapItem type element MUST be only in principal aliaser configuration files for regular expression principal aliasers. This section specifies only the differences between these elements and the type elements and child elements specified in section 2.1.

Each RegExMapItem type element specifies a different mapping of security principals using regular expressions. A file can contain multiple RegExMapItem type elements. Each RegExMapItem type element MUST have a unique name attribute which MUST be the value of exactly one name attribute in a dependency element of a RegExMap type element.

Throughout this section, XPath notation is used to specify nested XML elements, as specified in [XPATH] section 2. The paths are relative to the path named "/configuration/unity/containers/container".

2.3.1.4.1 Type Element

The type element MUST contain the attributes as specified in the following table.

<table>
<thead>
<tr>
<th>Attribute name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>A unique identifier that MUST match the name attribute of a dependency element specified in section 2.3.1.3.3.</td>
</tr>
</tbody>
</table>

2.3.1.4.2 Param Element

This specifies the param child elements of the type/typeConfig/constructor elements for a RegExMapItem type element. All name and parameterType attributes associated with the param element MUST contain the specified values in the specified order. Each child element MUST be as specified in the following table.

<table>
<thead>
<tr>
<th>Value of param name attribute</th>
<th>Value of param parameterType attribute</th>
<th>Child element</th>
</tr>
</thead>
<tbody>
<tr>
<td>log</td>
<td>ILog</td>
<td>MUST be a dependency element with no attributes.</td>
</tr>
<tr>
<td>Value of param name attribute</td>
<td>Value of param parameterType attribute</td>
<td>Child element</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>pattern</td>
<td>System.String</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute that specifies a regular expression pattern as specified in [MC-RegEx]. The pattern determines whether the user object or group object is mapped to another user store. The regular expression pattern can contain regular expression capture substrings that are used in the <code>&lt;param name=&quot;mapToUsername&quot;&gt;</code> element value.</td>
</tr>
<tr>
<td>mapToUsername</td>
<td>System.String</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute that specifies the <code>user name</code> or group to which to map. The <code>value</code> attribute can contain regular expression capture substitutions using the syntax &quot;#&quot; where # is the group capture index, as specified in [MC-RegEx].</td>
</tr>
<tr>
<td>inputPropertyName</td>
<td>System.String</td>
<td>MUST be a <code>value</code> element that contains a <code>value</code> attribute that specifies the name of the <code>property</code> attribute of a security principal that is input to the mapping. The regular expression pattern in the <code>&lt;param name=&quot;pattern&quot;&gt;</code> element MUST be the same as the value of the input property of a security principal. The <code>inputPropertyName</code> MUST be one of the following: $PRINCIPAL_REFERENCE_ID: The identifier of the user or group. This is the default. $PRINCIPAL_REFERENCE_ALIAS: The alias of the user or group. Any other property of the user or group. Other properties that the input user store exposes are specified with the user store.</td>
</tr>
</tbody>
</table>

2.3.2 Standard Principal Aliaser

There are two types of standard principal aliaser, the XML principal aliaser and the regular expression principal aliaser. Differences them are specified in the following sections.

2.3.2.1 XML Principal Aliaser

An XML principal aliaser configuration file MUST contain only the `XMLPrincipalAliaser type` element (section 2.3.1.1).

2.3.2.2 Regular Expression Principal Aliaser

A regular expression principal aliaser configuration file MUST contain the following `type` elements.

- One `RegExPrincipalAliaser type` element (section 2.3.1.2).
- One `RegExMap type` element (section 2.3.1.3).
- Any number of `RegExMapItem type` elements (section 2.3.1.4).
3 Structure Examples

The examples in this section are annotated with XML comments, "<!--comment-->", within the files themselves.

3.1 User Store Configuration File Format

The following examples describe user store configuration formats.

3.1.1 Claims User Store Configuration File

The following shows an example of a claims user store configuration file. The user store identifier of the user store is "clm" and the issuer is set to "@ALL@".

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!-- See section 2.1.7.1. -->
<configuration>
  <!-- See section 2.1.7.2. -->
  <configSections>
    <!-- See section 2.1.7.3. -->
  </configSections>
  <!-- See section 2.1.7.4. -->
  <unity>
    <!-- See section 2.1.7.5. -->
    <typeAliases>
      <!-- See section 2.1.7.6.1. -->
      <typeAlias alias="singleton" type="Microsoft.Practices.Unity.ContainerControlledLifetimeManager, Microsoft.Practices.Unity, version=1.2.0.0, PublicKeyToken=71e9bce11e9429c" />
      <!-- See section 2.1.7.6.2. -->
    </typeAliases>
    <!-- See section 2.1.7.7. -->
    <containers>
      <!-- See section 2.1.7.8. -->
      <container>
        <!-- See section 2.1.7.9. -->
        <types>
          <!-- See section 2.2.1.1. A Domain <type> element. -->
          <!-- See section 2.2.1.1.1. -->
            <!-- See section 2.1.7.11. The <lifetime> and <typeconfig> elements and their attributes are not specified in the Domain <type> element, so the common file format values in section 2.1 are used. -->
            <lifetime type="singleton" />
          </type>
        </types>
      </container>
    </containers>
  </unity>
</configuration>
```

[MS-FSSACFG] — v20120630
Search Authorization Configuration File Format

Copyright © 2012 Microsoft Corporation.

Release: July 16, 2012
<typeConfig
extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
Microsoft.Practices.Unity.Configuration, version=1.2.0.0,
PublicKeyToken=71e9bce11e9429c">
  <!-- See section 2.2.1.1.2. -->
  <constructor>
    <param name="logger" parameterType="ILog"
        <dependency />
    </param>
    <param name="id" parameterType="System.String">
      <!-- See section 2.2.1.1.2. As specified in the table, this line specifies the user store identifier. -->
      <value value="clm" />
    </param>
    <param name="description" parameterType="System.String">
      <!-- See section 2.2.1.1.2. As specified in the table, this line specifies a short description of the user store. -->
      <value value="description of claims user store" />
    </param>
    <param name="contentType" parameterType="System.String">
      <value value="Claims" />
    </param>
    <param name="preFilterGenerator"
      propertyType="Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator,
      <dependency name="SimplePreFilterGenerator" />
    </param>
  </constructor>
  <!-- See section 2.2.1.2. A PreFilterGeneratorSimple <type> element. -->
  <type name="SimplePreFilterGenerator"
    type="Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator,
    <!-- See section 2.1.7.11. The <lifetime> and <typeConfig> elements and their attributes are not specified in the PreFilterGeneratorSimple <type> element, so the common file format values in section 2.1 are used. -->
    <lifetime type="singleton" />
    <!-- See section 2.1.7.12. -->
  </typeConfig>
</typeConfig>
<dependency />
</param>
</constructor>
<!-- See section 2.2.1.2.3. -->
<property name="AllowPrefix" propertyType="System.String">
 <value value="docacl:" />
</property>
<property name="DenyPrefix" propertyType="System.String">
 <value value="docacl:9" />
</property>
<property name="NonIndexableCharacterPattern" propertyType="System.String">
 <value value="[^a-zA-Z0-9]" />
</property>
<property name="NonIndexableCharacterHandling" propertyType="System.String">
 <value value="encode" />
</property>
<property name="NonIndexableCharacterEncodingPrefix" propertyType="System.String">
 <value value="3nc0dd" />
</property>
</typeConfig>
</type>
<!-- See section 2.2.1.5. A ClaimsUserMonitor type element. -->
<!-- See section 2.2.1.5.1. -->
type name="ClaimsUserMonitor"
<!-- See section 2.1.7.11. The <lifetime> and <typeconfig> elements and their attributes are not specified in the ClaimsUserMonitor type element, so the common file format values in section 2.1 are used. -->
<lifetime type="singleton" />
<!-- See section 2.1.7.12. -->
typeConfig extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement, Microsoft.Practices.Unity.Configuration, version=1.2.0.0, PublicKeyToken=71e9bce111e9429c">
<!-- See section 2.2.1.5.2. -->
<constructor>
 <param name="log" parameterType="ILog">
  <dependency />
 </param>
 <param name="manager">
  <param name="log" parameterType="ILog">
  <dependency />
 </param>
  <param name="manager">
   <param name="log" parameterType="ILog">
   <dependency />
   </param>
  </param>
 </param>
</constructor>
</typeConfig>
3.1.2 Lotus Notes User Store Configuration File

The following is an example of a local cache user store configuration file.

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<configuration>
  <configSections>
  </configSections>
  <unity>
    <typeAliases>
      <typeAlias alias="singleton" type="Microsoft.Practices.Unity.ContainerControlledLifetimeManager, Microsoft.Practices.Unity, version=1.2.0.0, PublicKeyToken=71e9bce111e9429c" />
    </typeAliases>
    <containers>
      <container>
        <types>
        </types>
      </container>
    </containers>
  </unity>
</configuration>
```
extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
Microsoft.Practices.Unity.Configuration, version=1.2.0.0,
PublicKeyToken=71e9bce111e9429c">
<!-- See section 2.2.1.1.2. -->
<constructor>
  <param name="logger" parameterType="ILog">
    <dependency />
  </param>
  <param name="id" parameterType="System.String">
    <!-- See section 2.2.1.1.2. As specified in the table, this
    line specifies the user store identifier. -->
    <value value="lts" /></param>
  <param name="description" parameterType="System.String">
    <!-- See section 2.2.1.1.2. As specified in the table, this
    line specifies a short description of the user store.
    In this example, no description is provided. -->
  </param>
  <param name="contentType" parameterType="System.String">
    <value value="LotusNotes" /></param>
  <param name="preFilterGenerator" parameterType="Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator,
    <dependency name="LotusNotesPreFilterGenerator" />
  </param>
</constructor>
<!-- See section 2.2.1.3. A LotusNotesPreFilterGenerator <type>
element. -->
<property type="Microsoft.SharePoint.Search.Extended.Security.IUserMonitor,
  <dependency name="LocalCacheUserMonitor" />
</property>
</typeConfig>
<!-- See section 2.2.1.3.1. -->
type="Microsoft.SharePoint.Search.Extended.Security.IPreFilterGenerator,
Microsoft.SharePoint.Search_EXTENDED_SECURITY"
<!-- See section 2.1.7.11. The <lifetime> and <typeConfig>
elements and their attributes are not specified in the
PreFilterGeneratorSimple <type> element, so the common file
format values in section 2.1 are used. -->
lifetime type="singleton" />
<!-- See section 2.1.7.12. -->
typeConfig
extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
Microsoft.Practices.Unity.Configuration, version=1.2.0.0,
PublicKeyToken=71e9bce111e9429c">
<!-- See section 2.2.1.3.2. -->
<constructor>
  <param name="logger" parameterType="ILog">
    <dependency />
  </param>
</constructor>
</typeConfig>
<param name="useDenyOnDefault" parameterType="System.Boolean">
<!-- See section 2.2.1.3.2. As specified in the table, this line specifies the user security filter denies by default. -->
<value value="false" type="System.Boolean" />
</param>

<param name="useDocumentSecurity" parameterType="System.Boolean">
<!-- See section 2.2.1.3.2. As specified in the table, this line specifies if the collaborative business application protocol server is configured to use "document-level" security. -->
<value value="true" type="System.Boolean" />
</param>

<param name="useWildcardSecurity" parameterType="System.Boolean">
<!-- See section 2.2.1.3.2. As specified in the table, this line specifies if the collaborative business application protocol server is configured to use "wild card" security. -->
<value value="true" type="System.Boolean" />
</param>

<param name="useViewSecurity" parameterType="System.Boolean">
<!-- See section 2.2.1.3.2. As specified in the table, this line specifies if the collaborative business application protocol server is configured to use "view" security. -->
<value value="false" type="System.Boolean" />
</param>
</constructor>

<!-- See section 2.2.1.3.3. -->

<property name="AllowPrefix" propertyType="System.String">
<value value="docacl:" />
</property>

<property name="DenyPrefix" propertyType="System.String">
<value value="docacl:9" />
</property>

<property name="NonIndexableCharacterPattern" propertyType="System.String">
<value value=".*" />
</property>

<property name="NonIndexableCharacterHandling" propertyType="System.String">
<value value="encodeWithPrefix" />
</property>

<property name="NonIndexableCharacterEncodingPrefix" propertyType="System.String">
<value value="3nc0dd" />
</property>
</typeConfig>
</type>

<!-- See section 2.2.1.4. A LocalCacheUserMonitor <type> element -->

<!-- See section 2.2.1.4.1. -->
<!-- See section 2.1.7.11. The <lifetime> element and its attributes are not specified in the LocalCacheUserMonitor <type> element, so the common file format values in section 2.1 are used. -->
<lifetime type="singleton" />

<!-- See section 2.2.1.4.2. The <typeConfig> element and its attributes are specified in the LocalCacheUserMonitor <type> element, so the LocalCacheUserMonitor TypeConfig values specified in 2.2.1.4.2 are used instead of the values in 2.1.2.13.1. -->
</typeConfig>

<!-- See section 2.2.1.4.3. -->
<constructor>
  <param name="log" parameterType="ILog">
    <dependency />
  </param>
  <param name="largeFileStorage" parameterType="System.String">
    <dependency name="LargeStorePath" />
  </param>
  <param name="databaseFileName" parameterType="System.String">
    <!-- See section 2.2.1.4.3. As specified in the table, this line specifies the file name for the local cache storage file. -->
    <value value="lts.gum" />
  </param>
</constructor>

<!-- See section 2.2.1.4.3. -->
<property name="Name" propertyType="System.String">
  <value value="LocalCacheUserMonitor" />
</property>

<!-- See section 2.2.1.4.3. As specified in the table, this line specifies the initial capacity of the database file. -->
<property name="InitialCapacity" propertyType="System.Int32">
  <value value="5003" type="System.Int32" />
</property>

<!-- See section 2.2.1.4.3. As specified in the table, this line specifies the number of bytes for storing a security principal identifier. -->
<property name="IDSize" propertyType="System.Int32">
  <value value="100" type="System.Int32" />
</property>

<!-- See section 2.2.1.4.3. As specified in the table, this line specifies the number of bytes for storing a security principal alias. -->
<property name="NameSize" propertyType="System.Int32">
  <value value="150" type="System.Int32" />
</property>

<!-- See section 2.2.1.4.3. As specified in the table, this line specifies the maximum number of nested parent security principals a specific security principal can have. -->
<property name="MaxParents" propertyType="System.Int32">
  <value value="500" type="System.Int32" />
</property>
3.2 Principal Aliaser Configuration File Format

3.2.1 XML Principal Aliaser Configuration File

The following shows an XML aliaser configuration file that maps from the dm1 user store to the dm2 and dm3 user stores. The name of the mapping file is "xml_mappings_c9e1879f-6d62-459a-9200-24224531efa3.xml", and it uses the default input property value "$PRINCIPAL_REFERENCE_ID".

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<configuration>
  <!-- See section 2.1.7.1. -->
</configuration>
  <!-- See section 2.1.7.2. -->
<configSections>
  <!-- See section 2.1.7.3. -->
  <section name="unity"
    type="Microsoft.Practices.Unity.Configuration.UnityConfigurationSection,
    Microsoft.Practices.Unity.Configuration, Version=1.2.0.0,
    Culture=neutral,
    PublicKeyToken=71e9bce111e9429c" />
</configSections>
  <!-- See section 2.1.7.4. -->
<unity>
  <!-- See section 2.1.7.5. -->
<typeAliases>
  <!-- See section 2.1.7.6.1. -->
  <typeAlias alias="singleton"
    type="Microsoft.Practices.Unity.ContainerControlledLifetimeManager,
    Microsoft.Practices.Unity, version=1.2.0.0,
    PublicKeyToken=71e9bce111e9429c" />
  <!-- See section 2.1.7.6.2. -->
  <typeAlias alias="ILog"
</typeAliases>
  <!-- See section 2.1.7.7. -->
<containers>
  <!-- See section 2.1.7.8. -->
  <container>
    <!-- See section 2.1.7.9. -->
</container>
</containers>
</unity>
</configuration>
```
<!-- See section 2.3.1.1. An XMLPrincipalAliaser<type> element. -->
<!-- See section 2.3.1.1.1. -->
<type type="Microsoft.SharePoint.Search.Extended.Security.IPrincipalAliaser,
</type>

<!-- See section 2.1.7.11. The <lifetime> and <typeconfig> elements and their attributes are not specified in the
XMLPrincipalAliaser <type> element, so the common file format
values in section 2.1 are used. -->
<lifetime type="singleton" />
<!-- See section 2.1.7.12. -->
<typeConfig
  extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
               Microsoft.Practices.Unity.Configuration, version=1.2.0.0,
               PublicKeyToken=71e9bce111e9429c">
  <!-- See section 2.3.1.1.2. -->
  <constructor>
    <!-- See section 2.3.1.1.2. As specified in the table, this
         line specifies the principal aliase
         r identifier. -->
    <param name="logger" parameterType="ILog">
      <dependency />
    </param>
    <!-- See section 2.3.1.1.2. As specified in the table, this
         line specifies the input user store identifier. -->
    <param name="id" parameterType="System.String">
      <!-- See section 2.3.1.1.2. As specified in the table, this
           line specifies a list of output user store identifiers.
           -->
    <!-- See section 2.3.1.1.2. As specified in the table, this
         line specifies the name of the XML principal aliase
         mapping file. -->
    <param name="xmlFileName" parameterType="System.String">
      <!-- See section 2.3.1.1.2. As specified in the table, this
           line specifies the property of a security principal that
           is the input to the mapping. -->
      <value value="$PRINCIPAL_REFERENCE_ID" /></param>
  </constructor>
</typeConfig>
3.2.2 Regular Expression Principal Aliaser Configuration File

The following is an example of a principal alias configuration file which maps from the pri user store to the dm3 user store. It will do regular expression pattern matching to map the default input property matching ".(\.+)." to the user name \"\1\". It will map "djones" to "jonesd".

```xml
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<configuration>
  <configSections>
  </configSections>
  <unity>
    <typeAliases>
      <typeAlias alias="singleton" type="Microsoft.Practices.Unity.ContainerControlledLifetimeManager, Microsoft.Practices.Unity, version=1.2.0.0, PublicKeyToken=71e9bce111e9429c" />
    </typeAliases>
    <containers>
      <container>
        <types>
        </types>
      </container>
    </containers>
  </unity>
</configuration>
```

<!-- See section 2.1.7.11. The <lifetime> and <typeconfig> elements and their attributes are not specified in the Domain <type> element, so the common file format values in section 2.1 are used. -->
<lifetime type="singleton" />

<!-- See section 2.1.7.12. -->
<typeConfig
extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
Microsoft.Practices.Unity.Configuration,
version=1.2.0.0, PublicKeyToken=71e9bce11e9429c">
<constructor>
  <param name="logger" parameterType="ILog">
    <dependency />
  </param>
  <param name="id" parameterType="System.String">
    <!-- See section 2.3.1.2.2. As specified in the table, this line specifies the principal aliaser identifier. -->
    <value value="priToDm3RegEx" />
  </param>
  <param name="inputUserStoreId" parameterType="System.String">
    <!-- See section 2.3.1.2.2. As specified in the table, this line specifies the input user store identifier. -->
    <value value="pri" />
  </param>
  <param name="outputUserStoreIds" parameterType="System.String[]">
    <!-- See section 2.3.1.2.2. As specified in the table, this line specifies the list of output user store identifiers. -->
  </param>
    <dependency name="RegExMap" />
  </param>
</constructor>

<!-- See section 2.3.1.2.3. -->
<property name="CaseSensitive" propertyType="System.Boolean">
  <!-- See section 2.3.1.2.3. As specified in the table, this line specifies if the pattern matching is case sensitive. -->
  <value value="true" type="System.Boolean" />
</property>

<property name="UnicodeCaseSensitive" propertyType="System.Boolean">
  <!-- See section 2.3.1.2.3. As specified in the table, this line specifies if the pattern matching is Unicode case sensitive. -->
  <value value="true" type="System.Boolean" />
</property>

</typeConfig>
</type>

<!-- See section 2.3.1.3. A RegExMap <type> element. -->
<!-- See section 2.3.1.3.1. -->
type="RegExMap"
Extended.Security.WorkerLibrary.System.RegExPatternMap,
System.RegExPatternMap,
<!-- See section 2.1.7.11. The <lifetime> element and its
attributes are not specified in the RegExMap <type> element,
so the specified common file format values in section 2.1 are
used. -->
<lifetime type="singleton" />
<!-- See section 2.3.1.3.2. The <typeConfig> element and its
attributes are specified in the RegExMap <type> element, so
the RegExMap TypeConfig Element values specified in 2.3.1.3.2 are
used instead of the CT_Lifetime element values in 2.1.2.13.1. -->
?typeConfig
<!-- See section 2.3.1.3.3. The name attribute here matches the
name attribute in the following RegExMapItem <type> element. -->
<dependency name="RegExMapItem0" />
</typeConfig>
</type>
<!-- See section 2.3.1.4. A RegExMapItem <type> element. -->
<!-- See section 2.3.1.4.1. -->
type="RegExMapItem0"
<!-- See section 2.1.7.11. The <lifetime> and <typeconfig>
elements and their attributes are not specified in the
RegExMapItem <type> element, so the common file format values
in section 2.1 are used. -->
<lifetime type="singleton" />
<!-- See section 2.1.7.12. -->
?typeConfig
extensionType="Microsoft.Practices.Unity.Configuration.TypeInjectionElement,
Microsoft.Practices.Unity.Configuration, version=1.2.0.0,
PublicKeyToken=71e9bce111e9429c">
<!-- See section 2.3.1.4.2. -->
<constructor>
<param name="log" parameterType="ILog">
<dependency />
</param>
<param name="pattern" parameterType="System.String">
<!-- See section 2.3.1.4.2. As specified in the table, this
line specifies a regular expression pattern to match.
In this example, the first character is put into group
1 and the remaining characters are put into group2. -->
:value value="(.)(.+)">
</param>
<param name="mapToUsername" parameterType="System.String">
<!-- See section 2.3.1.4.2. As specified in the table, this
line specifies the output of the mapping. In this example, the output is group2 followed by group 1. It will turn "djones" into "jonesd".

```xml
<value value="\2\1" />
</param>
<param name="inputPropertyName" parameterType="System.String">
  <!-- See section 2.3.1.4.2. As specified in the table, this line specifies the property of a security principal that is the input to the mapping. -->
  <value value="$PRINCIPAL_REFERENCE_ID" />
</param>
</constructor>
</typeConfig>
</type>
</types>
<!-- See section 2.1.7.18 -->
<instances />
</container>
</containers>
</unity>
</configuration>
4 Security Considerations

4.1 User Store Configuration File Format

A user store configuration file contains the security sensitive information. The implementation that uses the file will secure it at all times.

4.2 Principal Aliaser Configuration File Format

A principal aliaser configuration file contains the security sensitive information. The implementation that uses the file will secure it at all times.
5 Appendix A: Full XML Schemas

For ease of implementation, the full XML schema is provided here. The user store configuration and the principal aliaser configuration file formats have the same schema.

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:complexType name="CT_ConfigSections">
  <xs:sequence>
    <xs:element name="section" type="CT_Section" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Configuration">
  <xs:sequence>
    <xs:element name="configSections" type="CT_ConfigSections" /> 
    <xs:element name="unity" type="CT_Unity" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Constructor">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="param" type="CT_Param" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Container">
  <xs:sequence>
    <xs:element name="types" type="CT_Types" />
    <xs:element name="instances" type="CT_Instances" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Containers">
  <xs:sequence>
    <xs:element name="container" type="CT_Container" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Dependency">
  <xs:attribute name="name" type="xs:string" use="optional" />
</xs:complexType>

<xs:complexType name="CT_Instances">
</xs:complexType>

<xs:complexType name="CT_Lifetime">
  <xs:attribute name="type" type="xs:string" use="required" fixed="singleton" />
</xs:complexType>

<xs:complexType name="CT_Param">
  <xs:choice>
    <xs:element minOccurs="0" name="value" type="CT_Value" />
    <xs:element minOccurs="0" name="dependency" type="CT_Dependency" />
  </xs:choice>
  <xs:attribute name="name" type="xs:string" use="required" />
</xs:complexType>

```
<xs:attribute name="parameterType" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="CT_Property">
  <xs:sequence>
    <xs:element minOccurs="0" name="value" type="CT_Value" />
    <xs:element minOccurs="0" name="dependency" type="CT_Dependency" />
  </xs:sequence>
  <xs:attribute name="name" type="xs:string" use="required" />
  <xs:attribute name="propertyType" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="CT_Section">
  <xs:attribute name="name" type="xs:string" use="required" fixed="unity" />
  <xs:attribute name="type" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="CT_Type">
  <xs:sequence>
    <xs:element name="lifetime" type="CT_Lifetime" />
    <xs:element name="typeConfig" type="CT_TypeConfig" />
  </xs:sequence>
  <xs:attribute name="type" type="xs:string" use="required" />
  <xs:attribute name="name" type="xs:string" use="optional" />
  <xs:attribute name="mapTo" type="xs:string" use="optional" />
</xs:complexType>

<xs:complexType name="CT_TypeAlias">
  <xs:attribute name="alias" type="xs:string" use="required" />
  <xs:attribute name="type" type="xs:string" use="required" />
</xs:complexType>

<xs:complexType name="CT_TypeAliases">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="typeAlias" type="CT_TypeAlias" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_TypeConfig">
  <xs:choice>
    <xs:element minOccurs="0" maxOccurs="unbounded" name="dependency" type="CT_Dependency" />
    <xs:element minOccurs="0" maxOccurs="unbounded" name="value" type="CT_Value" />
  </xs:sequence>
  <xs:element minOccurs="0" name="constructor" type="CT_Constructor" />
  <xs:element minOccurs="0" maxOccurs="unbounded" name="property" type="CT_Property" />
</xs:complexType>

<xs:complexType name="CT_Types">
  <xs:sequence>
    <xs:element maxOccurs="unbounded" name="type" type="CT_Type" />
  </xs:sequence>
</xs:complexType>
<xs:complexType>
  <xs:sequence>
    <xs:element name="typeAliases" type="CT_TypeAliases" />
    <xs:element name="containers" type="CT_Containers" />
  </xs:sequence>
</xs:complexType>

<xs:complexType name="CT_Value">
  <xs:attribute name="value" type="xs:string" use="optional" />
  <xs:attribute name="type" type="xs:string" use="optional" />
  <xs:attribute name="typeConverter" type="xs:string" use="optional" />
</xs:complexType>

<xs:element name="configuration" type="CT_Configuration" />
</xs:schema>
6 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- Microsoft® FAST™ Search Server 2010

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.
7 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.
Normative references 7

Overview (synopsis) 7

Principal aliaser configuration file format (section 1.4.2, section 1.5.2)

Principal aliaser configuration file format elements
- RegExMap 34
- RegExMapItem 35
- RegExPrincipalAliaser 31
- XMLPrincipalAliaser 30

Principal aliaser configuration file format example – regular expression 46

Principal aliaser configuration file format example – XML 44

Principal aliaser configuration file format standard principal aliasers
- regular expression 36
- XML 36

Principal aliaser configuration file format structure 29

Principal aliaser configuration file format type elements
- regular expression 36
- XML 36

Product behavior 54

References 6

informative 7

normative 7

RegExMap type element 34

RegExMapItem type element 35

RegExPrincipalAliaser type element 31

Regular expression principal aliasers 36

Relationship to protocols and other structures 8

Relationship to protocols and other structures – principal aliaser configuration file format 9

Relationship to protocols and other structures – user store configuration file format 8

Security
- principal aliaser configuration file format 50
- user store configuration file format 50

Simple types
- common file format 20

Standard principal aliasers
- regular expression 36
- XML 36

Standard user stores
- claims user store 29

local cache user store 29

User store configuration file format (section 1.4.1, section 1.5.1)

User store configuration file format elements
- ClaimsUserMonitor type 28
- domain type 20
- LocalCacheUserMonitor type 25
- LotusNotesPreFilterGenerator type 23
- PreFilterGeneratorSimple type 22

User Store Configuration File Format example 37

User store configuration file format example – claims 37

User store configuration file format example – Lotus Notes 40

User store configuration file format standard user stores
- claims 29
- local cache 29

User store configuration file format structure 20

User store configuration file format type elements 20

User stores
- claims user store 29
- local cache user store 29

Vendor-extendible fields 10

Versioning 10

XML principal aliasers 36

XML schema 51

XMLPrincipalAliaser type element 30