IT Connect

Get Help

UW Information Technology

TOOLS, SERVICES & SUPPORT

GUIDES BY TOPIC

GETTING STARTED

IT AT THE UW

GET HELP

Search IT Connect for:

/ Tools, Services & Support / IT Systems Infrastructure / Microsoft Infrastructure / Other IT Services and Help / Understanding SDDL Syntax

Microsoft Infrastructure

UW Azure

Authentication Integration

Communications

Entra ID (Azure Active Directory)

Getting Started: Delegated OUs

Getting Started: Windows Domains

Microsoft Infrastructure Service Design

Other IT Services and Help

– Bitlocker Recovery

– What is the Higher Education IT Environment?

– AD Security Overview

– Azure Virtual Desktop

– Frequently Asked Questions

– Microsoft Store for Business

– Understanding SDDL Syntax

– Unix Interop with NETID AD – Community contributions

– Windows Domain Setup at the UW

Software

Understanding SDDL Syntax

Last updated: April 10, 2024
Audience: IT Staff / Technical

What follows is an appendix which pieces together several disparate Microsoft documents on the SDDL syntax. The SDDL syntax is important if you do coding of directory security or manually edit a security template file.

SDDL (Security Descriptor Definition Language)

At the lowest level, the Security Descriptor Definition Language is used in the nTSecurityDescriptor attribute (and on registry keys and NTFS files) to define the ACL. Fortunately, one does not need to know this level of detail in normal conditions. But advanced administrators may want to write scripts or code that can correctly construct SDDL strings. Also the security templates (located at %systemroot%\security\templates\) use SDDL if you manually edit them with a text editor instead of the MMC interface. I've found that manually editing these templates turns out to be the most effective way to manipulate them. It is most likely that you will simply need to be able to read a

SDDL string, so we'll try to keep things to only a cursory overview. Further details can be found at http://msdn.microsoft.com/library/.

Format of nTSecurityDescriptor string (bold and italics added for clarity):

O:owner_sidG:group_sidD:dacl_flags(string_ace1)(string_ace2)…(string_acen)S:sacl_flags(string_ace1)(string_ace2)…(string_acen)

Each nTSecurityDescriptor SDDL string is composed of 5 primary parts which correspond to the Header, DACL (D:), SACL (S:), primary group (G:)and owner (O:). Each of these parts is designated with the prefix noted in parenthesis. The header contains some record keeping information, along with 2 flags that designate whether the object is blocking inheritance for the SACL and DACL. The contents of both the primary group and owner parts are simply a single SID. The contents of both the SACL and DACL parts are a string with no fixed length. ACEs make up the contents of these strings. ACEs are enclosed within parenthesis, and there are 6 fields in each ACE. These 6 fields are separated by a semicolon delimiter. The fields are ACE type (allow/deny/audit), ACE flags (inheritance and audit settings), Permissions (list of incremental permissions), ObjectType (GUID), Inherited Object Type (GUID), and Trustee (SID).

ACE Type

The ACE type designates whether the trustee is allowed, denied or audited.

Value	Description
"A"	ACCESS ALLOWED
"D"	ACCESS DENIED
"OA"	OBJECT ACCESS ALLOWED: ONLY APPLIES TO A SUBSET OF THE OBJECT(S).
"OD"	OBJECT ACCESS DENIED: ONLY APPLIES TO A SUBSET OF THE OBJECT(S).
"AU"	SYSTEM AUDIT

“AL”	SYSTEM ALARM
“OU”	OBJECT SYSTEM AUDIT
“OL”	OBJECT SYSTEM ALARM

ACE Flags

The ACE flags denote the inheritance options for the ACE, and if it is a SACL, the audit settings.

Value	Description
“CI”	CONTAINER INHERIT: Child objects that are containers, such as directories, inherit the ACE as an explicit ACE.
“OI”	OBJECT INHERIT: Child objects that are not containers inherit the ACE as an explicit ACE.
“NP”	NO PROPAGATE: ONLY IMMEDIATE CHILDREN INHERIT THIS ACE.
“IO”	INHERITANCE ONLY: ACE DOESN’T APPLY TO THIS OBJECT, BUT MAY AFFECT CHILDREN VIA INHERITANCE.
“ID”	ACE IS INHERITED
“SA”	SUCCESSFUL ACCESS AUDIT
“FA”	FAILED ACCESS AUDIT

Permissions

The Permissions are a list of the incremental permissions given (or denied/audited) to the trustee- these correspond to the permissions discussed earlier and are simply appended together. However, the incremental permissions are not the only permissions available. The table below lists all the permissions.

Value	Description	Hexadecimal Value	Binary Bits from 0
Generic access rights			
"GA"	GENERIC ALL	0x10000000	Bit 28
"GR"	GENERIC READ	0x80000000	Bit 31
"GW"	GENERIC WRITE	0x40000000	Bit 30
"GX"	GENERIC EXECUTE	0x20000000	Bit 29
Directory service access rights			
"RC"	Read Permissions	0x20000	Bit 17
"SD"	Delete	0x10000	Bit 16
"WD"	Modify Permissions	0x40000	Bit 18
"WO"	Modify Owner	0x80000	Bit 19
"RP"	Read All Properties	0x00000010	Bit 4
"WP"	Write All Properties	0x00000020	Bit 5
"CC"	Create All Child Objects	0x00000001	Bit 0

“DC”	Delete All Child Objects	0x00000002	Bit 1
“LC”	List Contents	0x00000004	Bit 2
“SW”	All Validated Writes	0x00000008	Bit 3
“LO”	List Object	0x00000080	Bit 7
“DT”	Delete Subtree	0x00000040	Bit 6
“CR”	All Extended Rights	0x00000100	Bit 8
File access rights			
“FA”	FILE ALL ACCESS		
“FR”	FILE GENERIC READ		
“FW”	FILE GENERIC WRITE		
“FX”	FILE GENERIC EXECUTE		
Registry key access rights			
“KA”	KEY ALL ACCESS	0xF003F	
“KR”	KEY READ	0x20019	
“KW”	KEY WRITE	0x20006	
“KX”	KEY EXECUTE	0x20019	

	KEY CREATE SUB KEYS	0x0004	
	KEY ENUMERATE SUB KEYS	0x0008	
	KEY NOTIFY	0x0010	
	KEY QUERY VALUE	0x0001	
	KEY SET VALUE	0x0002	

Object Type and Inherited Object Type

The ObjectType is a GUID representing an object class, attribute, attribute set, or extended right. If present it limits the ACE to the object the GUID represents. The Inherited Object Type is a GUID representing an object class. If present it limits inheritance of the ACE to the child entries of only that object class.

Trustee

The Trustee is the SID of the user or group being given access (or denied or audited). Instead of a SID, there are several commonly used acronyms for well-known SIDs. The most common are listed in the table below, but you can review more at https://docs.microsoft.com/en-us/windows/win32/secauthz/sid-strings:

Value	Description
“AO”	Account operators
“RU”	Alias to allow previous Windows 2000
“AN”	Anonymous logon
“AU”	Authenticated users

“BA”	Built-in administrators
“BG”	Built-in guests
“BO”	Backup operators
“BU”	Built-in users
“CA”	Certificate server administrators
“CG”	Creator group
“CO”	Creator owner
“DA”	Domain administrators
“DC”	Domain computers
“DD”	Domain controllers
“DG”	Domain guests
“DU”	Domain users
“EA”	Enterprise administrators
“ED”	Enterprise domain controllers
“WD”	Everyone
“PA”	Group Policy administrators

“IU”	Interactively logged-on user
“LA”	Local administrator
“LG”	Local guest
“LS”	Local service account
“SY”	Local system
“NU”	Network logon user
“NO”	Network configuration operators
“NS”	Network service account
“PO”	Printer operators
“PS”	Personal self
“PU”	Power users
“RS”	RAS servers group
“RD”	Terminal server users
“RE”	Replicator
“RC”	Restricted code
“SA”	Schema administrators

“SO”	Server operators
“SU”	Service logon user

Each of these fields and values has quite a bit of detail to it, and for the purposes of this overview we can’t go into extended detail.

Re-used with permission from Stanford University for which Brian Arkills originally wrote this documentation.

Is there a problem on this page? Let us know.

INFORMATION TECHNOLOGY
UNIVERSITY of WASHINGTON

Connect with UW-IT:

Get Help / My Requests / Service Status / About IT Connect / Sitemap / UW-IT Jobs / UW-IT Internal

Accessibility / Campus Safety / Workday / My UW / Privacy / Security / Terms