OpenVMS Technical Journal V14

HP OpenVMS CIFS File Security and Management

Shilpa K, HP OpenVMS CIFS File Security and Management

Intended Audience

This article is intended for OpenVMS administrators who are responsible for managing HP OpenVMS CIFS product. Within the scope of HP OpenVMS CIFS as Member Server, this article provides information about HP OpenVMS CIFS File Security. HP OpenVMS CIFS File Security is used for managing permissions for users and groups that are accessing files on an OpenVMS system through HP OpenVMS CIFS. This article also explains various aspects of HP OpenVMS CIFS File Security such as user, group and permission mapping.

Introduction to CIFS

HP OpenVMS CIFS is based on Open Source Samba. According to Samba.org, the definition of samba is "*Samba is an Open Source/Free Software suite that provides seamless file and print services to SMB/CIFS clients. Samba is freely available, unlike other SMB/CIFS implementations, and allows for interoperability between Linux/Unix servers and Windows-based clients. Samba is software that can be run on a platform other than Microsoft Windows, for example, UNIX, Linux, IBM System 390, OpenVMS, and other operating systems. Samba uses the TCP/IP protocol that is installed on the host server. When correctly configured, it allows that host to interact with a Microsoft Windows client or server as if it is a Windows file and print server.*"

Samba is based on Microsoft Common Internet File System (CIFS) protocol. CIFS protocol mainly uses Server Message Block (SMB) commands for communicating with different systems over network.

Samba being an Open Source product with a world-wide community of developers, it keeps pace with new Microsoft Operating System releases and thus provides seamless integration with Windows systems. To take advantage of this and thus to provide OpenVMS customers with file and print services that keep pace with new Windows releases, Open Source Samba has been ported onto OpenVMS. This is intended as an alternative to the existing file and printer services product, Advanced Server for OpenVMS. The ported version of Open Source Samba on OpenVMS is referred by the name, HP OpenVMS CIFS. HP OpenVMS CIFS is supported on OpenVMS version 8.2 and later on Alpha platform, and OpenVMS version 8.2-1 and later on Integrity servers.

The main features of HP OpenVMS CIFS henceforth referred simply as CIFS (not to be confused with CIFS protocol) are:

1. Domain Support
2. Authentication
3. Cluster Services
4. Browsing
5. File and Print Services
6. File and Print Security

A brief description about each of the main features and a detailed description of File Security are provided in the following sections.

Domain Support:
CIFS can act as a NT4-style Member Server in any domain. From CIFS version 1.2 onwards, it can participate as a member in native mode Active Directory Windows domain that uses Kerberos authentication.
It can act as a NT4-style Primary Domain Controller (PDC), but such a domain may only contain Backup Domain Controllers (BDCs) that run CIFS. Similarly, it can function as a NT4-style BDC only if the PDC is also running CIFS. However, unlike HP Advanced Server for OpenVMS and Windows domain controllers, automatic replication of the user accounts database is not possible between CIFS

PDC and BDCs. To accomplish the automatic replication of account databases, CIFS requires the assistance of LDAP servers. By configuring the CIFS PDC and BDCs to use the LDAP backend, replication of the accounts database is achieved by virtue of the synchronization between LDAP servers. CIFS can use the LDAP backend to store and obtain user and group account information in the LDAP directory (such as HP Enterprise Directory or an OpenLDAP server).

Though a single LDAP server can be used for both the CIFS PDC and BDCs, it is highly recommended that separate LDAP servers be used for high availability and better performance.

Authentication:
CIFS supports the basic and less secure share level security wherein the password is supplied when accessing each share, and a more secure user level security where the username and password must be supplied to successfully establish connection to CIFS Server before accessing shares.

In user level security, CIFS supports the following authentication mechanisms:

1. LM - used by old Windows systems
2. NTLM - used by Windows NT and later systems
3. NTLMv2 - used by Windows NT and later systems
4. Kerberos authentication from CIFS version 1.2 onwards (When CIFS acts as a Member Server in Windows Active Directory)

Additionally, CIFS provides:

1. NT LAN Manager Security Support Provider (NTLMSSP) support for securing NTLM and NTLMv2 authentication
2. Session security by signing and sealing secure channel data between a domain member and a domain controller. The CIFS session security can use 64-bit or128-bit encryption key for encrypting the secure channel data.
3. SMB signing or security signatures, which is used for securing SMB protocol.

Cluster Services:
CIFS can be installed either on a single node in an OpenVMS Cluster or, as a CIFS Member Server, on multiple nodes that share the same CIFS installation directory.

As a single node, CIFS can be installed as a distinct entity (for any CIFS server role) on each cluster node with separate installation and configuration and thus each node acts as if on non-clustered OpenVMS system. In this case, each node where CIFS is installed must not share the same installation directory and must not allow access to the same directory or files through multiple cluster members.

As a Member server, a common cluster configuration is possible where multiple cluster members can share the same CIFS installation and configuration directory and data files. In such an environment, CIFS functions as though the cluster is a single domain entity. In a cluster, the nodes that share the same CIFS installation and configuration directory must also share the same SYSUAF and RIGHTSLIST databases.

Browsing:
CIFS supports traditional Windows Browser service functionality. Browser service functionality is responsible for the “Network neighborhood” view provided by Windows.

File and Print Services:
CIFS allows users to share files and printers present on OpenVMS system to Windows, Linux, and UNIX clients. These clients see the shared files and printers as being present on a local system. Due to this, users can seamlessly work with shared files and printers using the available interfaces on the client system.

CIFS supports files present on ODS-5 and ODS-2 volumes. It can present files with different OpenVMS file formats and file organizations to Windows clients in a Stream format. Thus the files with different formats are made readable to Windows clients. Additionally, CIFS allows you to create files from clients in Stream, Stream_LF, Fixed, or Undefined format. By default, CIFS supports ASCII character set. Additionally, it supports Extended ASCII character set (CP850/ISO-8859-1) for some European characters and VTF-7 support for Japanese characters.

Using CIFS printing services, you can share printers that are connected directly to an OpenVMS system or any printers present on the network. For sharing printers using CIFS, print queues must be setup on the OpenVMS system. These print queues can be setup using DCPS, TELNETSYM, LAT, or LPD. CIFS supports NT-style printing functionalities such as:

- Printer driver files can be downloaded locally on Windows clients
- Printer driver files can be uploaded onto CIFS Server from the Windows clients using Add Printer wizard

File and Print Security:

An important requirement for any File and Print Services is file security. Unlike Advanced Server for OpenVMS, which provides NT ACL based file security as well as OpenVMS based file security, CIFS provides file security to Windows clients using OpenVMS file security alone. That is, it maps Windows security applied on the files and directories to OpenVMS file security. File security can be set for any Windows/CIFS user or group. File and directory access auditing is not provided by CIFS, but standard OpenVMS auditing can be used for this purpose.

CIFS supports Access Control Lists (ACLs) on printer objects as well.

CIFS Components

The main components of CIFS that provide the above features are:

1. SMBD process
2. NMBD process
3. WINBIND

SMBD process:

SMBD process is responsible for providing domain support, cluster services, authentication, File and Printer Services, and CIFS File Security mapping functionality. Each client session creates a new SMBD process. Each SMBD process provides domain support, cluster coordination services, authentication, file and print services, and File Security mapping.

NMBD process:

NMBD process provides traditional Windows Browser service functionality apart from handling NETBIOS name registration and resolution.

WINBIND

WINBIND is a special feature of CIFS that supports automatic mapping of Windows domain users and groups to OpenVMS UICs and resource identifiers, nested groups (a group with-in-a group) and trust functionality. In this article, winbind functionality and WINBIND are used interchangeably and they mean the same. The first 2 features of WINBIND will be covered in greater detail under the section “The Role of WINBIND in CIFS File Security”.

On other platforms like Linux, Samba provides winbind functionality through a process named, WINBINDD. On OpenVMS, winbind functionality is integrated into SMBD process.

The Scope:

Now that a brief summary of CIFS features and components have been provided, the scope of this article is limited to explaining File security when CIFS is configured as Member Server.
This article does not explain the steps for setting share ACLs on CIFS shares using Windows 'Computer Management' applet. The steps provided in this article are applicable for setting File Security on CIFS shares apart from directories and files under the CIFS shares.

CIFS as Member Server

As a member server, CIFS allows the following users to access CIFS shares:

1. Users belonging to the Windows domain, where CIFS is a member
2. Users belonging to the domains trusted by the Windows domain, where CIFS is a member
3. CIFS local users

A Windows domain user can be a member of multiple domain global groups in the Windows domain. The domain global groups can be added as members to CIFS local groups and this is referred as nested grouping. Users and domain global groups belonging to the domains trusted by the Windows domain, where CIFS is a member and CIFS local users and groups can also be a part of CIFS local groups.

CIFS allows you to set permissions on files and directories for the above mentioned users and groups.
You can also set permissions based on the mapped OpenVMS usernames and resource identifiers. This is supported because of the user and group mapping functionality provided by CIFS and due to the additional support it provides for OpenVMS File Security apart from Windows File security.

As mentioned under File and Print Security feature, CIFS maps Windows File Security to OpenVMS File Security. This raises the basic question: Why is CIFS providing this security mapping instead of providing Advanced Server style NT ACL support?

Why is CIFS Server dependent upon OpenVMS File Security?

For those customers who are planning to use CIFS, the likely concern about File Security is probably well expressed by the words in Samba HOW-TO collection on File Access Controls: "*Advanced MS Windows users are frequently perplexed when file, directory, and share manipulation of resources shared via Samba do not behave in the manner they might expect. MS Windows network administrators are often confused regarding network access controls and how to provide users with the access they need while protecting resources from unauthorized access.*

Many UNIX administrators are unfamiliar with the MS Windows environment and in particular have difficulty in visualizing what the MS Windows user wishes to achieve in attempts to set file and directory access permissions.

The problem lies in the differences in how file and directory permissions and controls work between the two environments. This difference is one that Samba cannot completely hide, even though it does try to bridge the chasm to a degree."

It continues to say *"This is an opportune point to mention that Samba was created to provide a means of interoperability and interchange of data between differing operating environments. Samba has no intent to change UNIX/Linux into a platform like MS Windows. Instead the purpose was and is to provide a sufficient level of exchange of data between the two environments. What is available today extends well beyond early plans and expectations, yet the gap continues to shrink."*

As part of porting Open Source Samba on to OpenVMS, due to the way File Security is implemented in Samba, CIFS had to map the Windows File Security to the nearest possible or an equivalent File Security provided by OpenVMS. To understand the File Security mapping provided by CIFS, it is necessary to know the following:

1. Windows File Security
2. OpenVMS File Security
3. The need for CIFS to map Windows domain users and groups to OpenVMS usernames and resource identifiers
4. User and group mapping
5. The role of WINBIND in CIFS File Security
6. Windows to OpenVMS File Security mapping provided by CIFS
7. Limitations due to mapping Windows File Security to OpenVMS File Security

Once these aspects are covered, this article further explains the steps required for setting up File Security – from a Windows system and from an OpenVMS system.

Note that the goal of this article is not to cover in detail about Windows and OpenVMS File Security. Only a brief introduction about these topics has been provided, which might help understand CIFS File Security better.

A brief introduction to Windows File Security

Windows File Security is mainly based on Security Principals and Discretionary Access Control Lists. A security principal can be a user or a group. On a Windows system, each user and group is identified by a unique Security Identifier (SID) in the domain. Windows creates an Access Token structure for a user after successfully authenticating the user. An Access Token structure consists of the logged in user's SID and the SIDs for each of the groups that the user belongs to. In Windows, a local group can contain another group within it and this is referred as nested-grouping. Suppose if a user ANITA belongs to group ACCOUNTS and if the group ACCOUNTS is a member of another group FINANCE, then by virtue of nested grouping, the user ANITA is a member of both ACCOUNTS and FINANCE groups.

Access to each file and directory (otherwise referred to as objects) in a Windows system can be restricted by applying permissions on them. These permissions on an object are applied at the discretion of an owner of the object and are also called Discretionary Access Control Lists (DACLs). The object's security descriptor is made up of these DACLs. The DACL is a list of Access Control Entries (ACE) on the object and each ACE contains permission for a user or group. Within the ACE, a user or group is represented by its SID. An ACE can be granted for users and groups in the same domain, for users and groups in the trusted domain and for users and groups in the local Windows system. Windows allows an explicit deny access to an object apart from allow access. Additionally, a user or a group can be granted explicit access rights (special privileges) by administrators that allow access to the object.

When a user tries to access an object, Windows decides access to the object based on the SIDs in the user's Access Token and the object's security descriptor consisting of ACEs. Windows grants or denies access rights by finding a match for the SIDs in the Access Token of the user with that of the object's ACE list. If a match is not found even after traversing the entire ACE list on the object, then access to the object is denied. In case the object's DACL contains more than one ACE that matches the SIDs in the user's Access Token, then Windows accumulates access rights for each ACE until the accumulated access rights exceeds the requested access rights. Thus, for a user ANITA, if the group ACCOUNTS allowed READ access to an object, and group FINANCE allowed WRITE access to the

same object, then Windows grants READ and WRITE access to the object for the user ANITA. From this, we can conclude that the order of ACEs in DACL of the object is not important on a Windows system.

A brief introduction to OpenVMS File Security

The OpenVMS operating system controls access to any object that contains shareable information. These objects are known as protected objects. Files and directories fall under the category of protected objects. An accessing process or thread carries credentials in the form of rights identifiers, and all protected objects list a set of access requirements specifying who has a right to access the object in a given manner. These are respectively known as User's Security Profile and Object's Security Profile.

User's Security Profile:
The types of identification the system assigns to processes to define their access rights to objects is referred to as the User's Security Profile. When a user tries to access an object, it is the process or thread executing on behalf of the user that uses the user's security profile to access the object. The User's Security Profile is also referred to as Subject's Security Profile. A User's Security Profile includes the following elements:

1. User identification code (UIC) identifying the user
2. Rights identifiers held by the process
3. Privileges, if any

User Identification Code (UIC)
The first element of a subject's security profile is the user identification code (UIC). Your UIC tells what system group you belong to and what your unique identification is within that group.

Rights Identifiers
The second element of a subject's security profile is a set of rights identifiers. A rights identifier represents an individual user or a group of users. While accessing files and directories, the Rights Identifiers of interest are:

1. General identifiers - Defined by the security administrator.
2. UIC identifiers - Based on a user's identification code (UIC), which uniquely identifies a user on the system and defines the group to which the user belongs.

Privileges
A third (optional) element of a subject's security profile is a set of privileges. Privileges let you use or perform system functions that ordinarily would be denied to you. The privileges held by the user can affect access to an object.

How Privileges Affect Protection Mechanisms
Security administrators can assign privileges to users when they create or modify user accounts. The system privileges READALL and BYPASS affect user access, regardless of the access dictated by an ACL for the object or by other elements in its security profile. The privileges SYSPRV and GRPPRV are controlled through the system category of the protection code. The privileges have the following meanings:

BYPASS	A user with BYPASS privilege receives all types of access to the object, regardless of its protection.
GRPPRV	A user with GRPPRV privilege whose UIC group matches the group of the owner of the object receives the same access accorded to users in the system category. Thus, the user with GRPPRV privilege is able to manage any of the group's objects.

READALL	A user with READALL privilege receives read access to the object, even if that access is denied by the ACL and the protection code. In addition, the user can receive any other access granted through the protection code.
SYSPRV	A user with SYSPRV privilege receives the access accorded to users in the system category.

When you define ACLs or protection codes for your objects, remember that users with amplified privileges are entitled to special access to objects throughout the system. For example, there is no way to stop a user with the BYPASS privilege from accessing your files. Users with GRPPRV privilege have the power to perform many system management functions for other members of their UIC group.

Object's Security Profile:
The previous section described User's Security Profile that is required for a user to access the object. The security elements of any object comprise its security profile and the object's security profile defines a list of requirements for accessing the object. An object's security profile contains the following types of information:

1. The owner of the object. The system uses this element in interpreting the protection code.
2. The protection code defining access to objects based on the categories of system, owner, group, and world. This protection code controls broad categories of users.
3. The access control list (ACL) controlling access to objects by individual users or groups of users.

A brief overview of security elements in Object Security Profile:

Owner
The first element of an object's security profile is the UIC of its owner. In most cases, if you create an object, you are its owner. As the owner, you can modify its security profile. The system automatically assigns your UIC to the object and uses it in making access decisions. There are some exceptions to the ownership rule. Files owned by resource identifiers do not have a UIC. When a user creates a file in the directory of a resource identifier, the file may be owned by the resource identifier and not the user who created the file.

Protection Code
The second element of an object's security profile is the object's protection code. The protection code associated with an object determines the type of access allowed to a user, based on the relationship between the user UIC and the owner UIC. A protection code defines the access rights for four categories of users: (a) the owner, (b) the users who share the same group UIC as the owner (the group category), (c) all users on the system (the world category), and (d) those with system privileges or rights (the system category). OpenVMS code lists access rights in a fixed order: the system category (S), then owner (O), then group (G), and then world (W). It has the following syntax: [user category: access allowed (,user category: access allowed,…)]

Access Control List (ACL)
The third (optional) element of an object's security profile is the object's access control list. An access control list (ACL) is a collection of Access Control Entries (ACEs) that define the access rights a user or group of users has to a particular protected object, such as a file, directory, or a device.

With this background information on User's Security Profile and Object's Security Profiles, the next section describes how the OpenVMS system determines a user's access to a protected object.

How the System Determines If a User Can Access a Protected Object

When a user tries to access a protected object, the operating system calls the Check Protection ($CHKPRO) system service to compare the security profile of the user process or thread with the security profile of the object. In the protection check, $CHKPRO compares the user's security profile against the protected object's profile using the following sequence:

1. Evaluate the access control list (ACL).
If the object has an ACL, the system scans it, looking for an entry that matches any of the user's rights identifiers. If a matching access control entry (ACE) is found, the system either grants or denies access, and further checking of the ACL stops.

When the matching ACE denies access, a user can still gain access either through the system and owner fields of the protection code or through privilege. When an ACL has no matching ACE, the system checks all fields of the protection code.

2. Evaluate the protection code.
If the ACL did not grant access and the object's owner UIC is not zero (refer NOTE), the operating system evaluates the protection code. The operating system grants or denies access based on the relationship between the user's identification code (UIC) and the object's protection code.
For cases where an ACL has denied access, the system examines two fields in the protection code---the system and owner fields---to determine if the user is allowed access. The user can still acquire access by being a member of the system or owner categories or by possessing privileges. A user holding GRPPRV (with a matching group UIC) or SYSPRV is granted the access specified for the system category of the protection code.

NOTE: When an object has an owner UIC of zero, the protection code is not checked. Users have all but control access to the object, *provided* the ACL has no Identifier ACEs. If Identifier ACEs are present, then access has to be granted explicitly through the ACL or through privilege.

3. Look for special privileges.
If access was not granted by the ACL or the protection code, privileges are evaluated.
Users with certain system privileges may be entitled to access regardless of the protection offered by the ACLs or the protection code. The bypass privilege (BYPASS), group privilege (GRPPRV), read all privilege (READALL), or system privilege (SYSPRV) amplifies the holder's access to objects.

The above three steps explains the fact that unless the user’s security profile has special privileges, and if the object has ACLs set on it, the order of the ACLs on the object’s security profile is very important in granting access to a user.

NOTE: For providing a brief description about OpenVMS File Security, almost all the information mentioned above has been directly obtained from the HP OpenVMS Guide to System Security. Minor modifications have been done to keep the security description to files and directories.

The need for user, group and permission mapping by CIFS

With the above information about Windows and OpenVMS File Security, it can be observed that there is no one-to-one mapping of Windows users and groups with OpenVMS usernames (UICs) and resource identifiers. As a File Server that uses CIFS or SMB protocol, CIFS is required to provide Windows-like File Security based on Windows users and groups. As CIFS is dependent upon host system File Security, it has to use OpenVMS File Security that is based on UICs and resource identifiers. The only way for CIFS to bridge the File Security on these two different Operating Systems is by mapping:

1. Windows users and groups to OpenVMS usernames (UICs) and resource identifiers
2. Windows permissions to OpenVMS permissions.

The following sections describe the Windows to OpenVMS user and group mapping and the permission mapping provided by CIFS.

User and Group mapping

Using the user and group mapping mechanism, CIFS maps domain users to OpenVMS usernames and domain groups to OpenVMS resource identifiers. Internally, the mapping is done between domain user SIDs and OpenVMS UICs and, domain group SIDs and OpenVMS resource identifier values. For easy readability, it is referred to as mapping of domain users and groups to OpenVMS usernames (UICs) and resource identifiers. The following topics in this section briefly describe the user and group mapping mechanism.

User Mapping

User mapping involves mapping users belonging to the Windows domain where CIFS is a member, the domains trusted by it and the CIFS server, to OpenVMS usernames (UICs). There are three ways using which CIFS maps these users to OpenVMS usernames:

1. Automatic user mapping provided by WINBIND.
2. Implicit user mapping – A domain user with the same username in SYSUAF on OpenVMS is implicitly mapped. CIFS local users are also mapped using this mechanism.
3. Explicit user mapping - Using username map file, domain users can be explicitly mapped to any OpenVMS username.

Automatic User Mapping

Users belonging to the Windows domain where CIFS is a member or the domains trusted by it can be automatically mapped to OpenVMS usernames provided that WINBIND is enabled and a valid idmap UID range is available in the Samba Configuration File. This will be explained in detailed under the topic 'The role of WINBIND in CIFS File Security'.

Implicit User Mapping

A user belonging to the Windows domain where CIFS is a member or the domains trusted by it can be implicitly mapped to an OpenVMS username provided the names are same. For example, when a user ANITA belonging to the Windows domain connects to CIFS Server, the user ANITA is implicitly mapped an OpenVMS username ANITA, if it exists.

By default, CIFS local users are implicitly mapped to OpenVMS usernames. This is because, a CIFS local user can be created using pdbedit utility, only if a matching OpenVMS username exists in SYSUAF on OpenVMS. For example, to create a CIFS local user STEFFI, you must first create or ensure that an OpenVMS username STEFFI exists in SYSUAF database. The CIFS user accounts database file, SAMBA$ROOT:[PRIVATE]PASSDB.TDB maintains records related to CIFS local users.

Explicit User Mapping - Mapping domain users using username map file

Explicit user mapping allows you to map users in the Windows domain where CIFS is a member and the domains trusted by it to any OpenVMS username using a username map file.

CIFS supports mapping of multiple domain users to a single OpenVMS username. In this case, the domain users that have been mapped to a single OpenVMS username share the File Security that has been specified for that mapped OpenVMS username. For example, if a directory and the sub directories and files under it are owned by a mapped OpenVMS username ASVUSER, then all the domain users that are mapped to OpenVMS username ASVUSER have access to this directory and the sub directories and files under it. The same is applicable if the mapped OpenVMS username ASVUSER has been explicitly granted resource identifiers in the SYSUAF database, and File Security

on any object in a CIFS share contains security for these resource identifiers. In this case, all the domain users that have been mapped to the OpenVMS username ASVUSER will also be able to access objects based on the resource identifiers that have been granted to the user ASVUSER in SYSUAF.

The following examples provide information on how to setup explicit user mapping:

Edit the username map file, SAMBA$ROOT:[LIB]USERNAME.MAP and add the user mappings in any of the following ways based on your CIFS setup requirements:

```
# The text following a hash (#) or a semi-colon (;) indicate comment lines.
# You can add your own comment lines by prefixing the text with a hash or a semi-colon

; Example for mapping a user in Windows domain (CIFSDOM) to OpenVMS username
SYSTEM=CIFSDOM\administrator

; Example for mapping multiple Windows domain users to a single OpenVMS username
ASVUSER=CIFSDOM\tunga CIFSDOM\kaveri

; Example for mapping a user in trusted domain (TRUSTDOM) to an OpenVMS username
CIFSUSER=TRUSTEDOM\neela

; An example of username map search stopping after encountering the required mapping
!GANGA=CIFSDOM\GANGES

; An example for mapping all the users connecting to CIFS to a single OpenVMS username
cifs$default=*
```

NOTE:

1. By default, the Samba configuration file parameter “username map” points to the CIFS supplied username map file, SAMBA$ROOT:[LIB]USERNAME.MAP
2. If you create your own “username map” file, ensure that it is in Stream or Stream_LF record format. Then you must update the Samba Configuration file parameter “username map” to point to your own username map file.
3. Do not use “cifs$default=*” unless you want to grant same File Security permissions to all the users connecting to CIFS.

Group Mapping

Group mapping in CIFS involves mapping domain global groups belonging to the domain where CIFS is a member, the domains trusted by the domain and the CIFS server, to OpenVMS resource identifiers.

The only supported mechanism for mapping global groups belonging to the domain where CIFS is a member and the domains trusted by it, to OpenVMS resource identifiers is the automatic mapping provided by WINBIND. This will be explained in detail under the section ‘The role of WINBIND in CIFS File Security’.

A CIFS local group can be explicitly mapped to an OpenVMS resource identifier using the command “NET GROUPMAP ADD”. This command automatically creates the CIFS local group as well as the required mapping with the OpenVMS resource identifier. The CIFS group mapping database file, SAMBA$ROOT:[VAR.LOCKS]GROUP_MAPPING.TDB stores the information about CIFS local groups,

members of CIFS local groups and the mapping between CIFS local groups and OpenVMS resource identifiers.

Summary

To summarize, CIFS local users and groups cannot be automatically mapped by WINBIND. Automatic mapping of users and groups by WINBIND is applicable only to users and groups in the Windows domain (where CIFS is a member) and the domains trusted by it.

The role of WINBIND in CIFS File Security

As mentioned earlier, WINBIND is a special feature of CIFS that provides the following functionalities:

1. Automatic Mapping - For domain users and groups, WINBIND automatically creates the corresponding OpenVMS users and groups (resource identifier) if a mapping for the same does not exist.
2. Nested Group Support – Using nested groups, domain global groups can be added to CIFS local groups (thus, a group-within-a-group, or "nested" groups). Nested groups are defined locally on any machine and can contain users and global groups from the domain (where CIFS is a member) and the domains trusted by it.
3. Trusts - WINBIND is required for all Trust functionality when CIFS is a PDC.

The winbind functionality provided by CIFS is controlled though the logical, WINBINDD_DONT_ENV. If it is disabled or not defined on a CIFS node, CIFS provides WINBIND support and if it is enabled by defining it to 1, CIFS turns off the WINBIND support. As this logical is not defined by default, CIFS provides WINBIND support by default. When CIFS is a Member Server, automatic mapping and nested group support provided by WINBIND play an important role in CIFS File Security. Due to this, it is recommended not to disable WINBIND support on a CIFS Member Server by defining the logical WINBINDD_DONT_ENV.

Though CIFS provides WINBIND support by default, the automatic mapping feature provided by WINBIND is enabled, only if you have specified a valid “idmap UID” and “idmap GID” range in the Samba Configuration File. The next topic ‘Automatic Mapping’ describes this in detail and the importance of automatic mapping in CIFS File Security.

Automatic mapping:

One of the purposes of WINBIND is to automate the creation of OpenVMS UICs and resource identifiers (in POSIX GID format) and maintain their correspondence to the appropriate Windows SIDs to minimize identity management efforts. The WINBIND identity mapping database file, SAMBA$ROOT:[VAR.LOCKS]WINBINDD_IDMAP.TDB maintains the mapping between Windows SIDs and OpenVMS UICs and resource identifiers. The mapping between a Windows SID and an OpenVMS username or a resource identifier is created only if there is no existing mapping entry in this database file. If the required mapping is missing, the automatic creation and mapping of OpenVMS usernames and groups (resource identifiers) occurs under the following two circumstances:

1. After the user is successfully authenticated, CIFS tries to map the authenticated user and all the groups that the user belongs, to an OpenVMS username (UIC) and groups (resource identifiers). The domain or CIFS groups can be either nested groups or the groups that the authenticated user directly belongs. If a mapping is found missing for an authenticated user and if this user is also a domain user, WINBIND can automatically create the required OpenVMS username and then map this OpenVMS username to the authenticated user. Similarly, if the mapping is found missing for any of the domain global groups that the user

belongs; WINBIND can create the required OpenVMS group (resource identifier) and map it to the domain group.

2. When setting security on files and directories in CIFS shares, if the security principal (subject) to whom you are granting the permission is a user or global group in the Windows domain (where CIFS is a member) or the domains trusted by it, WINBIND can create and map the required OpenVMS username (UIC) or resource identifier to a domain user or group SID.

When an OpenVMS username is created by WINBIND, it specifies a UIC for that username. Similarly, when it creates an OpenVMS resource identifier, it creates the identifier in POSIX group identifier format by supplying a GID value. In order for WINBIND to use the UIC and GID values specified by you while creating OpenVMS usernames and resource identifiers, WINBIND provides a mechanism, which allows you to explicitly specify a set of values for UICs and GIDs that are allocated solely to WINBIND. WINBIND obtains the user IDs (UIDs) used to assign a UIC value to an OpenVMS username, and group IDs (GIDs) used to assign a value to the POSIX group identifier (resource identifier) from the Samba configuration file global parameters "idmap UID" and "idmap GID". These parameters must be set to a range of values allocated solely to WINBIND. The next two sections explain the steps used by CIFS to create OpenVMS usernames and resource identifiers using “idmap UID” and “idmap GID” range of values.

How does WINBIND map domain users to OpenVMS users?

WINBIND uses the chosen integer "idmap UID" value, to derive both the OpenVMS username and the UIC. The UID value is converted to a hexadecimal value and appended to the string "CIFS$" to derive the OpenVMS username. The UID value is converted to octal and the octal value is used as the UIC group and member number.

NOTE: Since UIC group numbers are limited to a maximum value of Octal 37776 (decimal 16382), the upper range limit on the "idmap UID value is 16382. Similarly, because UIC group numbers below Octal 376 are reserved for use by HP, it is recommended not to specify a value below 255 as the lower range of "idmap UID".

For example, if the Samba configuration file contains:
idmap uid = 1000-2000

WINBIND will initially allocate UID 1000 and create an OpenVMS user named CIFS$3E8 with a UIC of [1750,1750]. The username CIFS$3E8 is created with interactive login disabled, nodisuser flag enabled and with NETMBX and TMPMBX privileges only. After the user CIFS$3E8 is successfully created in SYSUAF, the mapping of UID 1000 (for user CIFS$3E8) to a corresponding domain user SID is then stored in the file, SAMBA$ROOT:[VAR.LOCKS]WINBINDD_IDMAP.TDB. This file must be backed up regularly to avoid the loss of the required mappings necessary to maintain file security.

How does WINBIND map domain groups to OpenVMS resource identifiers?

WINBIND uses the chosen integer "idmap GID" value, to derive both the OpenVMS resource identifier (group) name and group identifier (GID) value. The GID value is converted to a hexadecimal value and appended to the string "CIFS$GRP" to derive the OpenVMS resource identifier name.

NOTE: Because WINBIND creates POSIX Group Resource Identifiers (POSIX GID), the maximum value is limited to %xFFFFFF or %d16777215. The lower limit is 1. OpenVMS automatically adds %xA4000000 to the value chosen. When CIFS is installed on an OpenVMS system, by default it automatically uses the GID values %xFFFFF0 thru %xFFFFFF and thus the highest GID value that can be specified is limited to %xFFFFFEF or %d16777199.

For example, if the SMB.CONF file contains:
idmap gid = 5000-10000

WINBIND will initially allocate GID 5000 and create an OpenVMS resource identifier named CIFS$GRP1388. After the resource identifier CIFS$GRP1388 is successfully created in the RIGHTSLIST database on OpenVMS, the mapping of GID 5000 (for group CIFS$GRP1388) to a corresponding domain group SID is then stored in the file, SAMBA$ROOT:[VAR.LOCKS]WINBINDD_IDMAP.TDB. It is critical that this file is backed up regularly as its loss will result in loss of the required mappings necessary to maintain security.

NOTE:

1. In an existing CIFS configuration, if you have to increase the "idmap uid" or "idmap gid" range values, retain the lower value of the range as it is while increasing only the higher value in the range. For example, in the "idmap uid" range specified above, to increase "idmap uid" range, specify the new range as "1000-3000". WINBIND will automatically adjust the existing "idmap uid" range while retaining the current mapping entries in the WINBINDD identity mapping database file. This guarantees that the existing security on files and directories that might have been set based on the existing mapping entries are still valid.
2. HP OpenVMS CIFS Administrator's Guide contains the flow charts on how the user and group mapping occurs in CIFS.
3. WINBIND automatic mapping feature is disabled by default as CIFS does not provide default values for "idmap uid" and "idmap gid" parameters. If WINBIND is enabled, automatic mapping feature is enabled once the valid "idmap uid" and "idmap gid" range values are specified in the Samba configuration file.
4. From CIFS V1.2 onwards, "idmap uid" and "idmap gid" ranges can be specified through Samba configuration utility.

Domain users and groups mapped by WINBIND:

WINBIND does not map all the users and global groups in the domain (where CIFS is a member) or in the domains trusted by it, even if automatic mapping is enabled. Instead it maps only the following domain users and groups:

1. Users belonging to the Windows domain (where CIFS is a member) or the domains trusted by it, who successfully established connection to CIFS Server at least once.
2. The domain global groups in the Windows domain where CIFS is a member or the domains trusted by it that contain the successfully authenticated users as members.
3. Users and groups belonging to the Windows domain where CIFS is a member and the domains trusted by it, to whom the CIFS administrator explicitly granted permissions for any Share, File or directory on CIFS.

Tracking and managing users and groups created by WINBIND

As explained above, WINBIND uses the range of values specified for "idmap uid" and "idmap gid" parameters in the Samba configuration file while creating OpenVMS usernames and resource identifiers. When WINBIND runs out of either of these idmap ranges that are specified in the Samba configuration file, it will report errors in the CIFS client log files. The client's log files will be created in the directory SAMBA$ROOT:[VAR].

CIFS provides a utility called WBINFO that can be used for viewing the OpenVMS usernames and resource identifiers created by WINBIND and the corresponding mapping to domain users and groups. You can execute the WBINFO utility as:

```
$ @SAMBA$ROOT:[BIN]SAMBA$DEFINE_COMMANDS.COM
$ WBINFO --hostusers-to-domainusers
$ WBINFO --hostgroups-to-domaingroups
```

The option "--hostusers-to-domainusers", displays the domain/CIFS users along with the mapped OpenVMS usernames. The option "--hostgroups-to-domaingroups", displays the domain/CIFS groups alongside the mapped OpenVMS resource identifiers. These two options are meant only for viewing the mapped users and groups and it cannot be used for enumerating the users and groups either in SYSUAF database on OpenVMS or in the domain/CIFS databases.

To check for the mapping between a single OpenVMS username or resource identifier (group) and a domain user or group, execute the command:

```
$ WBINFO --hostname-to-domainname=<OpenVMS username or resource identifier name>
```

When is automatic mapping provided by WINBIND required?

CIFS supports domain group to OpenVMS resource identifier mapping only through WINBIND. If you plan to set permissions on files and directories based on domain groups, automatic mapping provided by WINBIND is a must. Additionally, because WINBIND automates the OpenVMS username and resource identifier creation and the corresponding mapping with Windows SIDs, it can be used to avoid the manual administrative work related to identity management.

When is automatic mapping provided by WINBIND not required?

Automatic mapping provided by WINBIND need not be used if the following two conditions are satisfied:

1. All the domain users connecting to CIFS Server are either explicitly or implicitly mapped to the OpenVMS usernames.
2. Security Permissions on CIFS shares/files/directories are not based on domain global groups.

In this case, if you want to set permissions for domain global groups, then you must add them as members to CIFS local groups. Then, security on Files/Folders/Shares in CIFS can be set based on CIFS local groups. By virtue of nested group support provided by WINBIND, the security specified on an object for CIFS local groups that contain the domain global groups is automatically applicable to these domain global groups and the users belonging to them. The topic 'Managing CIFS local groups' provides examples on how to add domain users and domain global groups as members to CIFS local groups.

Nested group support:

As already mentioned, group within a group is referred to as nested grouping. Using the nested group support provided by WINBIND, you can add the following users and groups as members to CIFS local groups:

1. Users and domain global groups that belong to the Windows domain where CIFS is a member.
2. Users and domain global groups that belong to the domains trusted by the Windows domain where CIFS is a member.
3. Users and local groups in CIFS server database

As explained under "When is automatic mapping provided by WINBIND not required?" topic, Nested group functionality allows you to setup File Security based on CIFS local groups. WINBIND supports nested groups even when automatic mapping feature provided by it is disabled.

User Persona Creation

The mapping of domain/CIFS user and group names to OpenVMS UICs and resource identifiers occurs after the user connecting from a client system to CIFS server is successfully authenticated. After the user is authenticated, CIFS creates a persona for the user (User's Security Profile) that will be used by the SMBD process to access any object in CIFS server on behalf of the user. The persona for the authenticated user is internally created for the mapped OpenVMS username by the SMBD process.

The persona for the mapped OpenVMS user is made up of the following:

1. Mapped OpenVMS user's UIC and resource identifiers. These resource identifiers are the successfully mapped identifiers for the domain/CIFS groups that the authenticated user belongs.
2. Default privileges of the mapped OpenVMS username in SYSUAF.
3. Any general identifiers that were explicitly granted by the OpenVMS administrator for the mapped OpenVMS username in SYSUAF.

When an authenticated domain/CIFS user tries to access an object in CIFS share, OpenVMS grants access to the object based on the persona of the mapped OpenVMS user mentioned earlier. The only exception to this is, when the authenticated user is part of "admin users" in CIFS. When a user belongs to "admin users" in CIFS, the authenticated user's persona is same as that of the fully privileged user's persona that was originally created by the SMBD process for authenticating the domain/CIFS user. By default, the SMBD process authenticates the domain/CIFS users using the persona of SAMBA$SMBD account.

NOTE:

1. CIFS mandatorily requires that an identifier with the same UIC value as the UIC of the mapped OpenVMS username is present in the RIGHTSLIST database. For example, for a mapped OpenVMS user named, STEFFI with a UIC of [600,600], there must be a corresponding identifier with a UIC of [600,600].
2. Even though CIFS provides user mapping, a user connecting to the CIFS Server cannot be authenticated based on the credentials of the mapped OpenVMS username in SYSUAF.
3. SAMBA$SMBD account is created by CIFS as part of CIFS installation.

Windows to OpenVMS Permission Mapping provided by CIFS

The earlier sections explained the first part of CIFS File Security "User and Group Mapping". The following sections explain the Windows permissions to OpenVMS permissions mapping provided by CIFS.

Windows to OpenVMS Permission Mapping

OpenVMS allows you to specify READ (R), WRITE (W), EXECUTE (E), DELETE (D) or CONTROL (C) access permission on an object or a combination of RWEDC permissions (Refer note). On Windows, you can specify 'Full Control, Modify, Read and Execute, List Folders Contents (for directories only), Read, Write' standard permissions on an object using the 'Security' dialog box. Using the 'Advanced Security Setting' dialog box, Windows provides special permission setting. "Table 1 - Windows Permissions to OpenVMS permissions mapping" gives the list of Windows permissions that are mapped to corresponding OpenVMS permissions by CIFS. The "Advanced" in brackets indicates that this permission is available on Windows, only under the 'Advanced Security Setting' dialog box. 'Full Control' and 'Read' permissions are part of standard as well as special permissions.

Table 1 – Windows Permissions to OpenVMS Permission Mapping

Windows Permissions	**OpenVMS Permissions**
Full Control	RWEDC
Modify	RWED
Read and Execute	RE
Read	R
Write	W
Full Control (*Advanced*)	RWEDC
Traverse Folder / Execute File (*Advanced*)	E
List Folder / Read Data (*Advanced*)	R
Read Attributes (*Advanced*)	R
Read Extended Attributes (*Advanced*)	R
Create Files / Write Data (*Advanced*)	W
Create Folder / Append Data *(Advanced)*	W
Write Attributes *(Advanced)*	W
Write Extended Attributes *(Advanced)*	W
Delete Subfolders and Files (*Advanced*)	Not supported
Delete (*Advanced*)	D
Read Permissions (*Advanced*)	R
Change Permissions (*Advanced*)	C
Take Ownership (*Advanced*)	C

NOTE: Though the access permission, ACCESS=NONE specified on an object on OpenVMS is honoured by CIFS as it depends on OpenVMS to grant access, CIFS does not support setting this access permission on an object from a Windows system.

Windows Inheritance Value to OpenVMS inheritance mapping

On a Windows system, when setting permissions on a directory, you can specify if the access is applicable only to that directory or to the subfolders and files that will be created under it or to a combination of these.

Similarly, OpenVMS provides DEFAULT ACLs for the directories that are applicable only to the files and directories created under it while the access to the directory is controlled by the access ACE on the directory. "Table 2 – Windows inheritance value to OpenVMS inheritance mapping" provides information about Windows to OpenVMS inheritance mapping provided by CIFS.

Table 2 - Windows inheritance value to OpenVMS inheritance mapping

Windows Inheritance Value	**VMS Inheritance mapping**
This Folder only	Maps to access ACE
This Folder, Subfolders and Files	An ACE of this type is mapped to both access and default ACE.
This Folder and Subfolders	Maps *only* to access ACE for this directory.
This Folder and Files	Maps *only* to access ACE for this directory.

Subfolders and Files only	Maps to default ACE for this directory.
Subfolders only	This type is not supported and any ACE with this type is ignored by the CIFS.
Files only	This type is not supported and any ACE with this type is ignored by the CIFS.

Mapping OpenVMS RMS protection code to Windows Permissions

By default, OpenVMS automatically sets the protection code (RMS protection code) on each object when it is created. Though the ACLs on an object are optional, the protection code is a must. The syntax of the protection code consists of permissions for SYSTEM, OWNER, GROUP, and WORLD categories. Optionally, OpenVMS lets you specify inheritable DEFAULT_PROTECTION ACE on the directories that consists of protection code for these categories. The protection code specified in this ACE will be applied to the newly created files within the directory and the newly created directories within the parent directory will inherit this ACE.

On a Windows system, it has OWNER category to indicate the owner of the object and Everyone group that contains all the users on the Windows system. CREATOR OWNER and CREATOR GROUP present on the parent directory specify the permissions that will be inherited by the child objects for OWNER and the primary group of the owner respectively.

"Table 3 – OpenVMS RMS protection code category mapping to Windows mapping" shows the mapping provided by CIFS for OpenVMS protection code category to corresponding Windows mapping.

Table 3 - OpenVMS RMS protection code category mapping to Windows mapping

RMS Protection code category	**Windows mapping**
Owner/SYSTEM	Owner
Group	(displayed as) Unix Group
World	Everyone
Owner of default protection ACE	CREATOR OWNER on the directory
Group of default protection ACE	CREATOR GROUP on the directory
World of default protection ACE	Everyone for Subfolder and Files

Controlling OpenVMS Protection code using SMB.CONF parameters

CIFS provides various Samba Configuration File (SMB.CONF) parameters that allow you to control OpenVMS RMS protection code setting when:

1. The directories and files are newly created
2. Security permissions are explicitly set on the directories.

From CIFS patch set PS009 for CIFS V1.1 ECO1 onwards, the way these SMB.CONF parameters control OpenVMS RMS protection code have been simplified. Due to this, the implementation of many SMB.CONF parameters related to File Security is quite different on CIFS for OpenVMS when compared to Open Source versions of Samba (which are primarily based on the UNIX security model).

With this simplification, CIFS now allows OpenVMS to determine security using standard security rules when new files and directories are created. CIFS then adjusts the RMS protection code and

owner based on various SMB.CONF parameters; however, the defaults for these parameters are such that they will not modify the security that OpenVMS would apply (with exceptions noted).

The following SMB.CONF parameters can be used to modify security applied by OpenVMS:

1. inherit owner - The default value is "no". If set to "yes", causes CIFS to set the RMS owner of new objects to that of the parent directory.

NOTE: If "inherit owner = no" and a parent directory is owned by a Resource Identifier, when a non-privileged user who has WRITE access to this directory, creates a new file, CIFS sets the Owner to the UIC of the user creating the file, rather than the Resource Identifier. Thus, in order to retain the OpenVMS behavior (i.e., to set the resource identifier as owner), add "inherit owner = yes" to the applicable [share] sections of the SMB.CONF file. This will be addressed in a future release.

2. inherit vms rms protections - This is a new parameter with a default value of "no". If set to "yes", causes CIFS to:
    a. Set the RMS protection code to that of parent directory.
    b. Ignore a DEFAULT_PROTECTION ACE, if present.
    c. Ignore the RMS protection mask specified by the SYSGEN parameter RMS_FILEPROT.
    d. Ignore the mask and mode parameter values specified in the SMB.CONF file.

3. “Table 4 – mask and mode parameters” provides the list of mask and mode parameters supported by CIFS. The value of the appropriate "mask" parameter is AND'd with the result of the RMS protection code that OpenVMS would apply. This result is then OR'd with the value of the appropriate mode parameter:

Table 4 – mask and mode parameters

Parameter Name	**Affects RMS protection code when**
create mask	Creating new files
force create mode	Creating new files
directory mask	Creating new directories
force directory mode	Creating new directories
directory security mask	Windows users modify security on the directory
force directory security mode	Windows users modify security on the directory

NOTE:

i. CIFS for OpenVMS does not use the SMB.CONF parameters "security mask" and "force security mode" when a Windows user modifies the security of a file.
ii. As "create mode" parameter is same as "create mask" parameter, if required, use "create mask" parameter instead of "create mode".
iii. The following mask and mode parameter are related for doing AND & OR operation to generate a resultant RMS protection code:
    a. "create mask" and "force create mode"
    b. "directory mask" and "force directory mode"
    c. "directory security mask" and "force directory security mode"
iv. By default, the value of the mask parameters is 07777 and that of mode parameters is 00000 with the only exception of “force directory mode”. From CIFS version 1.2 onwards, the default value for “force directory mode” is 04000. This is to allow DELETE permission for OWNER category of the protection code when a new directory is created.

SMB.CONF parameters that cannot be modified

The following SMB.CONF parameters must not be modified from their default values and if you modify the value to a non-default value, it is not supported by CIFS:

1. inherit acls - Default is "yes"; you cannot disable inheritance of ACLs
2. inherit permissions - Replaced by the "inherit vms rms protections" parameter
3. security mask - Not supported
4. force security mode - Not supported

Delete access support for RMS protection code when using mask and mode parameters

One of the significant changes introduced with the release of patch set PS006 for CIFS V1.1 ECO1 concerns granting DELETE access in the RMS protection code on new objects. Previously, the various mask and mode parameters tied DELETE access to the WRITE bit; i.e. if you enabled WRITE access you also enabled DELETE access. However, with the release of PS006, DELETE and WRITE protections have been separated as documented below.

In addition, with the release of patch set PS009 for CIFS V1.1 ECO1, the default values for the mask and mode parameters have been changed such that they will not adjust the security that OpenVMS would itself apply (except where noted).

The values for the mask and mode parameters now have this significance:

<mask or mode parameter name> = 0dogw

Where:

'0' = Indicates the value is Octal
'd' = Controls granting DELETE access across all categories of the RMS protection code (see below)
'o' = Controls granting READ, WRITE, and EXECUTE access for the Owner category of the RMS protection code
'g' = Controls granting READ, WRITE, and EXECUTE access for the Group category of the RMS protection code
'w' = Controls granting READ, WRITE, and EXECUTE access for the World category of the RMS protection code

NOTE: The System category of the RMS protection code receives the same permission as the Owner category; there is no option to modify this behavior.

DELETE access is signified using a bitmask with the following values:

4 = Grant DELETE access to Owner category of RMS protection code
2 = Grant DELETE access to Group category of RMS protection code
1 = Grant DELETE access to World category of RMS protection code

The Owner, Group, and World access values are also bitmasks which signify the following access:

4 = Grant READ access
2 = Grant WRITE access
1 = Grant EXECUTE access

ACL order while applying CIFS File Security

In OpenVMS File Security section, it showed that the order in which the ACLs are applied on the files and directories is very important on an OpenVMS system while the same does not hold for Windows systems.

Due to the contrasting nature of Windows and OpenVMS systems' ACL processing, it is important to understand the order in which the ACLs are applied by CIFS when setting security on an object. When a CIFS administrator sets security permission on an object present in the CIFS share from a Windows system, CIFS first converts these Windows permissions to OpenVMS permissions. If the security was applied for users and groups, the converted OpenVMS permissions will contain OpenVMS ACLs for the mapped OpenVMS usernames and resource identifiers. After this conversion, CIFS applies the OpenVMS ACLs on the object. While applying OpenVMS ACLs on an object, CIFS must preserve the existing OpenVMS specific ACLs that would have been added by an OpenVMS administrator and these ACLs should ideally be retained in their original order (Refer note for information about OpenVMS specific ACEs). For example, on an OpenVMS system, users who fail to have access to a file based on any of the top order ACEs could be granted READ access by using an ACE, (IDENTIFIER=*,ACCESS=READ). Typically, an ACE similar to this would be present at the bottom of ACLs. CIFS, when setting permission, needs to ensure that this ACE is almost always present at the bottom of the ACLs on an object. Due to this reason, following design is implemented in CIFS when applying security on an object from a Windows system:

1. When a new ACE for a user or group is applied on an object in a CIFS share from a Windows system, after converting the Windows ACE to OpenVMS ACE format, CIFS applies this ACE at the top of the OpenVMS ACL order on an object.
2. While modifying an existing ACE for a user or group on an object in a CIFS share from a Windows system, after converting the Windows ACE to OpenVMS ACE format, CIFS finds out the location of the existing ACE that is getting modified. Then, it replaces the existing ACE with a modified ACE entry at the same location.
3. All the OpenVMS specific ACEs like AUDIT, ALARM, IDENTIFIER=*, PROTECTED and HIDDEN are retained in their existing locations.

NOTE: OpenVMS specific ACEs like AUDIT, ALARM, IDENTIFIER=*, PROTECTED, and HIDDEN that exist on an object in a CIFS share cannot be viewed or modified from a Windows system.

Limitations due to File Security Mapping

Until now, the article explained how the Windows File Security is mapped to OpenVMS File Security. This mapping mechanism is not without its limitations as Windows and OpenVMS systems vastly differ in the way they process ACLs on an object to grant access to the object. The limitations due to File Security mapping provided by CIFS are in the following areas:

1. Object access
2. File permission setting
3. Built-in Administrators group
4. Windows inheritance value mapping
5. Windows permission mapping
6. Viewing permissions on a directory from Windows

Object access limitation:

As mentioned in Windows File Security, Windows systems allow access to an object based on the accumulated access permissions for an object unless the user has special rights. Refer section 'Windows File Security' to understand how Windows derives accumulated access permissions for an

object. Thus the order in which the ACLs appear on an object has no importance on Windows systems.

On OpenVMS systems, the order in which the ACLs appear on an object is very important. This is because OpenVMS grants access to an object based on the first matching ACE that it encounters from the top of ACE list unless the user has special privileges or is an owner of the file.

When a user tries to access an object in a CIFS share from a client system, CIFS lets the OpenVMS system to decide access to the user based on the persona of the mapped OpenVMS user that was created by CIFS. According to the OpenVMS security rules, if the user is not an owner of the object and has no special privileges, user connecting to CIFS would be granted access to an object based on the first matching ACE encountered by OpenVMS system on that object. This can lead to access permission limitation in the following scenario:
A user ANITA belongs to two domain global groups FINANCE and ACCOUNTS. On an object, an ACE corresponding to one of the domain global groups FINANCE is granted READ access and is above the ACE corresponding to another domain global group ACCOUNTS which has full permissions (RWEDC). When a user ANITA that belongs to these 2 domain global groups access this object on a CIFS share, OpenVMS grants read only access to this user for the object based on the first matching ACE. On this object, the first matching ACE for the user ANITA is for the domain global group FINANCE. Thus, the ACE corresponding to the domain global group ACCOUNTS which has higher access permissions is not exercised for the user ANITA when the object is accessed by this user. This is an architectural limitation.

The workaround to this problem is to set security on an object in a CIFS share path from an OpenVMS system. While doing so, it must be ensured that the ACEs with highest access permissions are at the top of ACL order and the ACEs with lowest access permissions are at the bottom of ACL order.

File Permission setting limitation:

CIFS does not support exclusive permission setting on a file, though the same is supported for directories/folders. For example, in a directory with 100 files and folders under it, one user may have READ access to this directory and the files and folders under it. Just for a single file in this directory, you may want to grant modify (RWED) access for this user. This is not supported by CIFS with one exception. The exception to this problem is, either you make this user as OWNER of the file or grant 'Change Permission' (CONTROL) access to the file for this user. You may want to use this workaround only if you want this user to have control access to this file and not otherwise.

CIFS does not support exclusively specified OpenVMS specific ACEs (like PROTECTED, HIDDEN etc) on a file though the same is supported for directories. For example, in a directory with 100 files and folders, you might have exclusively set an AUDIT ACE on a particular file in this directory. A user with modify (RWED) access to this file, opens it, then saves the modified file and closes it. The AUDIT ACE that was exclusively set on this file might have been lost after the file was closed. This is particularly applicable for Microsoft office application files. The only workaround to this problem is to apply an inheritable default ACEs on the parent directory. This will lead to all the files and sub directories under the parent directory inheriting the ACE.

Built-in Administrators group limitation:

Prior to CIFS patch set PS005 for CIFS V1.1 ECO1, all the members of CIFS built-in (local) Administrators group were granted the two special OpenVMS privileges, BYPASS and SYSPRV. This allowed full administrative privileges to users belonging to built-in 'Administrators' group. By virtue of nested grouping, if the built-in 'Administrators' contained any other CIFS local groups or domain

global groups (like 'Domain Admins'), the users belonging to these nested groups were also granted the BYPASS and SYSPRV privileges. Due to this, these users were automatically entitled to perform CIFS Administrative work like File Security setting, user and group management and so on. From CIFS patch set PS005 for CIFS V1.1 ECO1 onwards, members of built-in Administrators group are no longer granted BYPASS and SYSPRV privilege. As a result, though the members of built-in 'Administrators' group can perform rest of the administrative work, they cannot set security on files and folders. Next section describes how the members belonging to built-in 'Administrators' group can be granted permission on files and directories for managing security on the same.

Windows inheritance value mapping limitation

When setting permissions on a directory in CIFS share from a Windows system, Windows provides multiple options on how the permissions can be applied on a directory. CIFS does not support 'Subfolder only' and 'Files only' Windows inheritance value options. Refer to "Table 2 - Windows inheritance value to OpenVMS inheritance mapping" for interpretation of rest of the Windows inheritance value options and their mapping to OpenVMS ACEs.

Windows permissions mapping limitation

CIFS does not support the special permission 'Delete Subfolder and Files' options that is available from the permissions list in the 'Advanced Security Setting' dialog box. If you try to set permission for this option, you are likely to encounter "Access denied" error.

Limitation in viewing permissions on a directory from Windows

From Windows, when viewing permissions on a CIFS directory or share using the 'Security' dialog box, you will see empty permissions for users and groups. This does not correctly reflect the permissions existing for the users and groups on that directory or share. In order to correctly view permissions on a CIFS directory or share, always use 'Advanced Security Setting' dialog box. In the 'Security' dialog box, click on 'Advanced' tab to go to 'Advanced Security Setting' dialog box.

On the other hand, you can correctly view permissions on files in a CIFS folder using the 'Security' dialog box itself.

Permissions and Privileges required by a user for setting up CIFS File Security

A user who tries to setup File Security on an object in a CIFS share must have certain privileges or permissions to successfully set up File Security. This section describes the permissions and privileges that allow a user to set up CIFS File Security.

Permissions for users belonging to built-in Administrators group:

The topic 'Built-in Administrators group limitation' mentioned the reasons why a user who belongs to built-in Administrators either directly or through nested grouping, can no longer set File Security or access files and directories in CIFS shares. CIFS provides an alternative option such that all the users belonging to the built-in Administrators group either directly or through nested grouping, can still manage File Security on files and directories in a CIFS share. To use this option, you should grant READ access to the files and directories in CIFS shares for the OpenVMS resource identifier, CIFS$ADMINISTRATORS. You can apply this identifier either for all the files and directories under all the CIFS share paths or only to a selected files and directories under certain share paths. Once the resource identifier CIFS$ADMINISTRATORS is applied on an object with READ access, while setting

permissions from a Windows system, a user belonging to built-in Administrators group (either directly or through nested grouping) can take ownership of the object. After the user becomes an owner of the object, the user is allowed to set required permissions for other users and groups on that object.

The READ access to the OpenVMS resource identifier, CIFS$ADMINISTRATORS can be granted by executing the commands similar to the following on an OpenVMS system:

```
$ SET DEFAULT [CIFSSHARE]
```

To apply inheritable default ACE to directories under CIFSSHARE.DIR, execute:

```
$ SET SECURITY/ACL= (IDENTIFIER=CIFS$ADMINISTRATORS,OPTIONS=DEFAULT,ACCESS=READ) –
_$  [...]*.DIR
```

To apply access ACE for files and directories under CIFSSHARE.DIR, execute:

```
$ SET SECURITY/ACL= (IDENTIFIER=CIFS$ADMINISTRATORS,ACCESS=READ) [...]*.*;*
```

If you would like to grant CIFS$ADMINISTRATORS resource identifier to the parent directory CIFSSHARE.DIR, execute:

Applying default ACE for inheritance:

```
$ SET SECURITY/ACL= (IDENTIFIER=CIFS$ADMINISTRATORS,OPTIONS=DEFAULT,ACCESS=READ) –
_$ [-]CIFSSHARE.DIR
```

Applying access ACE:

```
$ SET SECURITY/ACL= (IDENTIFIER=CIFS$ADMINISTRATORS,ACCESS=READ) [-]CIFSSHARE.DIR
```

NOTE:

1. CIFS, by default creates CIFS$ADMINISTRATORS OpenVMS resource identifier after CIFS has been successfully configured and connection was established to it at least once.
2. You can grant any other access permission to CIFS$ADMINISTRATORS in addition to READ depending upon the security setup in your CIFS environment.

Full privileges to selected domain users and CIFS local users:

For granting full administrative privileges to a selected list of domain users and CIFS local users, you can use the SMB.CONF parameter, “admin users”. By administrative privilege, it is meant full privileges that can be granted to a user on an OpenVMS system. You can follow the steps mentioned below for granting administrative privilege to a selected list of users.

On a CIFS member server, to grant an administrative privilege to a user belonging to the Windows domain (where CIFS is a member) or the domains trusted by it, the format of the “admin users” parameter in the Samba configuration file is:

```
admin users = <DOMAINNAME\domain-user>
```

On CIFS a member server, to grant an administrative privilege to a CIFS local user, the format of the “admin users” parameter in the Samba configuration file is:

```
admin users = <CIFS local username>
```

NOTE:

1. You can specify multiple usernames in the same line. When specifying multiple usernames, separate each username by a comma.
2. Domain usernames and CIFS local usernames can be specified in the same line. For example, domain users and CIFS local users can be specified in the same line as:

   admin users = <DOMAINNAME\domain-user>, <CIFS local username>

3. Prior to CIFS version 1.2, the "admin users" parameter can be added in the [global] section of the Samba configuration file SAMBA$ROOT:[LIB]SMB.CONF. From CIFS version 1.2 onwards, you can update the "admin users" parameter in the Samba configuration include file, SAMBA$ROOT:[LIB]ADMIN_USERS_SMB.CONF.
4. "admin users" parameter, if specified under the [global] section of Samba configuration file grants full privileges to all the CIFS shares and files and folders under these shares in addition to allowing these admin users the right to do other administrative operations
5. "admin users" parameter, if specified under the share section of the Samba configuration file, grants administrative privileges to the specified admin users only for that share.

Granting privileges to a mapped OpenVMS username

A domain or a CIFS local user whose account is mapped to an OpenVMS username will derive the same privileges as the mapped OpenVMS username. Due to this, a domain or a CIFS local user can become a privileged administrative user, provided you have granted the necessary privileges (like BYPASS, SYSPRV, GRPPRV) for the mapped OpenVMS username. The privileges mentioned here are the default privileges of a mapped OpenVMS username in SYSUAF database. CIFS does not honor authorized privileges of a mapped OpenVMS username.

You can use the following DCL command to grant privileges for a mapped OpenVMS user:

$ MCR AUTHORIZE MODIFY <OpenVMS-username>/DEFPRIVILEGES = (<privileges>)

Privileges can be BYPASS, SYSPRV, and GRPPRV etc. Multiple privileges can be specified by separating them using a comma.

NOTE: Some of the utilities provided by CIFS honor current privileges of the terminal process that is executing the utilities on behalf of the user.

Special permission – 'Change Permission' or CONTROL access

A user, who has 'Change Permission' or CONTROL access to an object, can modify permissions on that object for other users and groups. The same is possible, if any of the groups that the user belongs to (either directly or through nested grouping), has 'Change Permission' or CONTROL access to an object. Again, this is subject to OpenVMS security rules and depends on the ACE location in the ACLs on the object unless the user is an owner of the object or has special privileges.

By granting a 'Change Permission' (CONTROL access) on an object for a user or to any of the groups that the user belongs to, you allow the user to set permissions on this object for any other users and groups.

Privileges/Permissions for a user when setting CIFS File Security from an OpenVMS system

A user who tries to set up CIFS File Security on an object in a CIFS share path from an OpenVMS system must have full privileges (BYPASS privilege is a must) to set security.

Steps for setting up CIFS File Security

This section describes the steps that are required for setting up File Security on CIFS shares and directories. For setting File Security on a share using the steps provided below, the share is treated as a directory. This section includes the following topics:

1. Managing CIFS local groups – Describes how to add users and groups to CIFS local groups
2. Setting up File Security from a Windows system – Describes how to set up File Security from a Windows system.
3. Setting up File Security from an OpenVMS system - Describes how to set up File Security from an OpenVMS system.

NOTE: Before you follow rest of the topics, you must ensure that you have successfully added CIFS as member server to a Windows domain as rest of the topics assume that a working CIFS configuration is already in place.

As the above mentioned topics will be based on practical examples in most cases, sample configuration information used in these examples is as follows:

Windows Domain name (where CIFS is a Member Server):	CIFSDOM
CIFS Member Server name (OpenVMS system name):	PIANO
Windows PDC emulator name for the domain CIFSDOM:	ROX3

Managing CIFS local groups

This section describes the steps and commands that can be executed for managing CIFS local groups. Following users and groups can be added as members to a CIFS local group:

1. Users and global groups of the domain where CIFS is a member
2. Users and global groups of the domains trusted by the domain where CIFS is a member
3. CIFS local users and groups

NOTE: In an OpenVMS cluster, if multiple nodes in a cluster share the same samba$root installation directory where CIFS is configured as Member Server, use the CIFS cluster alias name for the "-W" option of NET RPC GROUP command instead of OpenVMS system name.

The steps for managing CIFS local groups are as follows:

Step 1: Login to an OpenVMS system and define Samba commands

1. Login to an OpenVMS system (PIANO) where CIFS is configured, using a fully privileged OpenVMS user account (say, SYSTEM).

2. Define Samba commands by executing:

   $ @SAMBA$ROOT:[BIN]SAMBA$DEFINE_COMMANDS.COM

Step 2: Creating a privileged CIFS user

To execute the commands mentioned in 'Step 3' that let you manage CIFS local groups, you must first create a CIFS local user with an administrative privilege if no such user exists. To do this, execute the following commands:

1. Create an OpenVMS username say, CIFSADMIN:

   $ SHOW LOGICAL SYSUAF

   If it is not defined, execute:
   $ DEFINE SYSUAF SYS$SYSTEM:SYSUAF

   $ MCR AUTHORIZE COPY SAMBA$TMPLT CIFSADMIN/UIC=[600,600]/FLAG=NODISUSER

   NOTE: From CIFS version 1.2 onwards, CIFS by default creates CIFSADMIN account in SYSUAF.

2. Create a CIFS user account with the same name as the OpenVMS username created in previous step:

   $ PDBEDIT "-a" CIFSADMIN
   new password: any1willdo
   retype new password: any1willdo

3. Update "admin users" parameter
   a. For versions prior to CIFS V1.2, edit the Samba configuration file, SAMBA$ROOT:[LIB]SMB.CONF and add or update (append) the following parameter in the [global] section:
      admin users = cifsadmin
   b. From CIFS version 1.2 onwards, edit the Samba configuration include file, SAMBA$ROOT:[LIB]ADMIN_USERS_SMB.CONF and update (append) the following parameter:
      admin users = cifsadmin

Step 3: Executing commands to add/modify/delete CIFS local groups

1. Adding or Creating a CIFS local group:

   a. Add a resource identifier say, CIFSUSERS in the RIGHTSLIST database by executing the following command:

   $ MCR AUTHORIZE ADD/IDENTIFIER/ATTRIBUTE=RESOURCE CIFSUSERS

   b. To add/create a CIFS local group by mapping it to the resource identifier, CIFSUSER, execute:

   $ NET GROUPMAP ADD NTGROUP=CIFSUSERS UNIXGROUP=CIFSUSERS TYPE="L"

2. To list the groups in CIFS local database, enter one of the following commands:

   $ NET GROUPMAP LIST

   OR

   $ NET RPC GROUP LIST "-W" PIANO "-S" PIANO "-U" CIFSADMIN%"Pwd of CIFSADMIN"

3. To add an already existing CIFS local user (STEFFI) or a local group (PLAYERS) to a CIFS local group (CIFSUSERS), enter the following commands:

   ```
   $ NET RPC GROUP ADDMEM CIFSUSERS STEFFI "-W" PIANO "-S" PIANO -
   _$ "-U" CIFSADMIN%"Pwd of CIFSADMIN"

   $ NET RPC GROUP ADDMEM CIFSUSERS PLAYERS "-W" PIANO "-S" PIANO -
   _$ "-U" CIFSADMIN%"Pwd of CIFSADMIN"
   ```

4. To a CIFS local group (CIFSUSERS), to add a domain user (ANITA) or a domain global group (ACCOUNTS) that belong to the domain CIFSDOM, enter the following commands:

   ```
   $ NET RPC GROUP ADDMEM CIFSUSERS CIFSDOM\ANITA "-W" PIANO –
   _$ "-S" PIANO "-U" CIFSADMIN%"Pwd of CIFSADMIN"

   $ NET RPC GROUP ADDMEM CIFSUSERS CIFSDOM\ACCOUNTS "-W" PIANO –
   _$ "-S" PIANO "-U" CIFSADMIN%"Pwd of CIFSADMIN"
   ```

5. To list members of a CIFS local group (CIFSUSERS), enter the following command:

   ```
   $ NET RPC GROUP MEMBERS CIFSUSERS "-W" PIANO "-S" PIANO –
   _$ "-U" CIFSADMIN%"Pwd of CIFSADMIN"
   ```

6. To delete a group from a CIFS local group (CIFSUSERS), enter the following command:

   ```
   $ NET RPC GROUP DELMEM CIFSUSERS CIFSDOM\ACCOUNTS "-W" PIANO –
   _$ "-S" PIANO "-U" CIFSADMIN%"Pwd of CIFSADMIN"
   ```

   NOTE: The same command can be used to delete a CIFS local user or a group OR a domain user or a domain global group. To delete a CIFS local user or a group, specify only the name, i.e., without the 'DOMAINNAME\'

7. To delete a CIFS local group (CIFSUSERS), enter the following command:

   ```
   $ NET RPC GROUP DELETE CIFSUSERS "-W" PIANO "-S" PIANO –
   _$ "-U" CIFSADMIN%"Pwd of CIFSADMIN"
   ```

   NOTE: You must first delete all the members from CIFS local group (CIFSUSERS) before deleting the group itself.

Additional Note:

In the above examples, to add/remove users and groups that belong to domains trusted by a Windows domain (CIFSDOM) to a CIFS local group, use the member name as 'TRUSTEDDOMAIN\<USERNAME or GROUPNAME>

Setting up File Security from a Windows system

This section describes the steps that should be followed for setting up File Security on shares/folders from a Windows system. It is assumed that you will be using a user account that has the required permissions or privileges (described in earlier sections) to execute the following steps.

Step 1: Login to CIFS Server

From the Windows system, connect to CIFS server (PIANO) by specifying \\<CIFS server name or IP address> at the prompt, "Start->Run" or map the required CIFS share by right-clicking on the 'My Computer' icon on the desktop and then select 'Map Network drive' in the menu. This will take you to 'Map Network drive' dialog box. In the 'Map Network Drive' dialog box, enter shared folder name under 'Folder' drop–down and then click 'Finish'. If prompted for credentials, connect using a user account (say, ANITA) that has the required privileges/permissions to manage the security on the shared folder.

Step 2: Setting security

a. Select the shared folder or any folder within the shared folder that you would like to manage and right-click on it and select Properties->security->Advanced.
b. If the user account (ANITA) that you used for connecting to CIFS Server in step 1 (under this topic) is a member of built-in Administrators group (either directly or because of nested grouping) and you are managing the share/folder only because of the permissions that have been granted for the OpenVMS resource identifier CIFS$ADMINISTRATORS, then follow rest of the steps under 'b', otherwise go to step 'c'.
    i. Select 'Owner' tab in the 'Advanced Security Setting' dialog box.
    ii. When the 'Owner' dialog box is displayed, check if you are the owner of this object. If so, proceed to step 'c'.
    iii. When the 'Owner' dialog box is displayed, if the connected user account name is not displayed in 'Change owner to' list, click on 'Other Users or Groups'. This will take you to 'Select User, Computer, or Group' dialog box.
    iv. In the 'Select User, Computer, or Group' dialog box, if connected user (ANITA) does not belong to the location (domain) displayed, click on 'Locations' tab. In the 'Locations' list box, select the appropriate location where the connected user account (ANITA) is present and click 'OK'.
    v. In 'Select User, Computer, or Group' dialog box, enter the connected user name (ANITA) under the 'Enter the object name to select (examples):' and click on 'Check Names'. If Windows finds multiple names for the name that you have entered, it will display 'Multiple Names Found' list box. In this box, select the appropriate name (RDN) and click 'OK'. If 'Multiple Names Found' box is not displayed or once you return from the 'Multiple Names Found' box, click 'OK' in the 'Select User, Computer, or Group' dialog box.
    vi. After returning to 'Owner' dialog box, select the connected user name (ANITA) under 'Change owner to' list box and click 'Apply' button. Once the owner name is set to connect user name (ANITA), select 'Permissions' tab. As you are now an owner of this object, proceed to step 'c' for setting permissions to other users and groups.
c. In the 'Advanced Security Setting' box, choose permissions tab (by default permissions tab is chosen). Click on "Add" tab if you would like add a new permission entry for a user or group, or to modify/remove an existing permission entry, select the user or group under the 'Permission entries'.
d. If you want to remove the selected 'Permission entry', click on 'Remove' tab and then click on 'Apply' button. Now, go to step 'i'.
e. If you want to modify the selected 'Permission entry', skip this step and go to step 'f'. If you want add a new 'Permission entry', follow rest of the steps under this step.
    i. If you want to add a new permission entry, click on the 'Add' tab. Windows now displays 'Select User, Computer, or Group' dialog box.
    ii. In the 'Select User, Computer, or Group' dialog box, if a user or group name to whom you want to grant permissions belongs to the displayed location (domain), proceed to next step 'iii'. If a user or group belongs to any other location (domain), click on 'Locations' tab; Select the appropriate 'Location' in the 'Locations' list box and click 'OK'.

iii. In 'Select User, Computer, or Group' dialog box, under 'Enter the object name to select (examples)', type the user or group name to whom you want to grant permission, Then, click on 'Check Names' tab. If Windows finds multiple names for the name that you have entered, it will display 'Multiple Names Found' list box. In this box, select the appropriate name (RDN) and click 'OK'. If 'Multiple Names Found' box is not displayed or once you return from 'Multiple Names Found' box, click 'OK' in the 'Select User, Computer, or Group' dialog box. From this step onwards, follow instructions from step 'f'.

f. Windows will now display 'Object' dialog box. Select the permissions that you would like to grant for the selected 'Name' from the 'Permissions' list. If you are setting permissions for a directory or a shared folder object, select the appropriate inheritance value from the 'Apply onto' drop-down. Next, click 'OK'.
g. Once you return to 'Permissions' dialog box in 'Advanced Security Setting', click on 'Apply' button.
h. Next, you can either repeat steps 'b' to 'g' or click on 'OK' to exit the 'Advanced Security Setting' dialog box. Once you return to 'Security' dialog box, click 'OK' to exit the 'Security' dialog box.

Setting up File Security from an OpenVMS system:

This section describes the steps/commands that should be followed for setting File Security from an OpenVMS system for users and groups belonging to the Windows domain, where CIFS is a member (CIFSDOM), the domains trusted by it and the CIFS Server. This method is limited to a user who has a fully privileged user account on the OpenVMS system. The steps to be followed are:

1. Login to an OpenVMS system using a fully privileged OpenVMS account, say "SYSTEM".

2. Define Samba commands by executing:

   $ @SAMBA$ROOT:[BIN]SAMBA$DEFINE_COMMANDS.COM

3. Find out the mapped OpenVMS identifier of a domain/CIFS user or group to whom you are trying to grant permissions by executing the following command and note down the identifier name (VMSIDENTIFIER-NAME) displayed:

   $ WBINFO –domainname-to-hostname=<DOMAINNAME\USERNAME OR GROUPNAME> VMSIDENTIFIER-NAME

   NOTE:
   a. If it is a user or a group belonging to Windows domain (CIFSDOM), where CIFS is a member, replace DOMAINNAME with the Windows domain name (CIFSDOM).
   b. If it is a user or a group belonging to a domain trusted by Windows domain (CIFSDOM), replace DOMAINNAME with the trusted domain name.
   c. If it is a user or a group that exists on the CIFS local database, replace DOMAINNAME with CIFS server name (PIANO).

4. Execute the following command to set required permission for the identifier (VMSIDENTIFIER-NAME) noted down in step '3'. For example,
   a. if you want to grant READ and EXECUTE access permissions to a directory DIRECTORY_NAME.DIR that is under a CIFS share path, you can execute:

   $ SET SECURITY/ACL=(IDENTIFIER=VMSIDENTIFIER-NAME,ACCESS=READ+EXECUTE) –
   _$ DEVICENAME:[PARENT_DIR_PATH]DIRECTORY_NAME.DIR

b. If you would like to apply the inheritable DEFAULT ACE on this directory, you can set DEFAULT ACE by executing:

```
$ SET SECURITY/ACL=(IDENTIFIER=VMSIDENTIFIER-NAME,OPTIONS=DEFAULT, -
_$ ACCESS=READ+EXECUTE) DEVICENAME:[PARENT_DIR_PATH]DIRECTORY_NAME.DIR
```

5. To modify permissions for an existing ACE and to replace the modified ACE at the existing ACE location, you can execute:

```
$ SET SECURITY/ACL=(IDENTIFIER=VMSIDENTIFIER-NAME,ACCESS=READ+EXECUTE) –
_$ /REPLACE= (IDENTIFIER=VMSIDENTIFIER-NAME,ACCESS=READ+WRITE+EXECUTE) -
_$ DEVICENAME:[PARENT_DIR_PATH]DIRECTORY_NAME.DIR
```

6. To insert an ACE after another ACE, you can execute:

```
$ SET SECURITY/ACL=(IDENTIFIER=VMSIDENTIFIER-NAME,ACCESS=READ+WRITE+EXECUTE) –
_$ /AFTER=(IDENTIFIER=EXISTING-VMSID-NAME,ACCESS=READ+WRITE+EXECUTE+DELETE) -
_$ DEVICENAME:[PARENT_DIR_PATH]DIRECTORY_NAME.DIR
```

   NOTE: The ACE specified for /AFTER must exactly match the existing ACE on the directory.

   NOTE: This step can be used as a workaround for the “File access limitation”. The ACE with higher permission can be place before the ACE with lower permission.

7. To remove an ACE for a user or a group whose identifier was obtained in ‘3’, execute the following command:

```
$ SET SECURITY/ACL=(IDENTIFIER=VMSIDENTIFIER-NAME, -
_$ ACCESS=READ+WRITE+EXECUTE)/DELETE –
_$ DEVICENAME:[PARENT_DIR_PATH]DIRECTORY_NAME.DIR
```

NOTE: Execute $ HELP SHOW SECURITY to obtain more help.

Backing up CIFS DATABASE files

The previous sections covered user and group mapping, permission mapping, and the steps that can be used for setting permissions. It can be noted that the permission on the objects in a CIFS share is applied based on the mapped OpenVMS usernames and resource identifiers (groups) and not based on the domain/CIFS users and groups that they are mapped to. From this, it should be noted as to how important it is to maintain this user and group mapping information for retaining security on the objects. As already mentioned in previous sections, few of the CIFS databases maintain this mapping information. The following databases in CIFS are crucial and they must be backed up regularly in order to avoid any loss of data due to avoidable or unavoidable reasons:

winbindd_idmap.tdb	- Stores records for WINBIND created mapping between domain SIDs and idmap UID/GID values (or OpenVMS UICs and resource identifiers).
group_mapping.tdb	- Stores records for CIFS local groups and the members of these groups
passdb.tdb	- Stores records for CIFS local users when passdb backend = tdbsam
secrets.tdb	- Stores private information like workstation passwords, the ldap admin dn and trust account information
account_policy.tdb	- Stores NT account policy settings such as pw expiration
share_info.tdb	- Stores information about share ACLs
ntdrivers.tdb	- Stores information about installed Printer drivers

ntforms.tdb - Stores information about installed Printer forms
ntprinters.tdb - Stores information about installed printers

The passdb.tdb and secrets.tdb database files will be present in the directory, SAMBA$ROOT:[PRIVATE]. Rest of the database files will be present in the directory, SAMBA$ROOT:[VAR.LOCKS].

For more information

The information present in this article has the following sources:
SAMBA Official HOW-TO
HP OpenVMS CIFS Administrator's Guide
HP OpenVMS Guide to System Security
Microsoft Authentication and Access Control Technologies
Release notes for CIFS Patch set for CIFS V1.1 ECO1