Creating User-Specific Files for OpenVMS Users AD Authentication Setup

Table of Contents

1. Introduction, Prerequisites, and Scope

This document details the creation of user/system-specific files to be used by OpenVMS during the User Active Directory (AD) Authentication setup. (Refer to the User Note: *AD Setup*.)

All prerequisites mentioned in *AD Setup* are relevant to this document. While that document generally describes AD Authentication setup, this document is intended to explain just a single step (the second step) of that setup; that is creating setup files (the development phase performed by HSQ engineers based on customer supplied information).

This means that the complete set of information collected from the customer during the first setup step is another development stage prerequisite. The necessary information must contain:

- The IP address (DNS name) of an authenticating server.
- The distinguished name of a proxy user account to be used by the Lightweight Directory Access Protocol (LDAP) service.
- The password for the above proxy account.
- The location in the directory under which the authentication records are kept.
- The list of usernames to be externally authenticated. This list must contain two usernames for every user account: Windows (AD) username and OpenVMS username. This can be the same or different for each user.

All files created using the instructions below should be protected and their security attributes be set to: **RWED,RE,RE,RE**.

NOTE: AD_SETUP.COM, **AD_SETUP_LOCAL.COM**, **AD_NODE_SETUP.COM**, **ACME$START.COM**, **SYS$LOGIN_SWITCH.COM**, **SYSTARTUP_VMS.COM**, **LDAP_ACCOUNTOFF.COM**, **AD_NEWUSER.COM**, **AD_UNINSTALL.COM** as well as all ***.*_TEMPLATE** files are not customer-dependent. Therefore, they can be a permanent part of the MISER distribution and always ready for use when needed.

In fact only two or three files are actually customer specific:

- The LDAP ACME configuration file (**LDAPACME$CONFIG-STD.INI**)
- The user accounts modification file (**LDAP_ACCOUNTMODIFY.COM**)
- The LDAP username database (**LDAP_LOCALUSER_DATABASE.TXT**) – if necessary

2. Creating the LDAP ACME Configuration File

The configuration file creation starts by utilizing the existing configuration file template:
SYS$STARTUP:LDAPACME$CONFIG-STD.INI_TEMPLATE

This file should be copied to create a working copy in a user-selected “development” location (e.g., the user’s home directory) and named “LDAPACME$CONFIG-STD.INI”. However, the name is not important, but in order to automate the setup as much as possible, the filenames are enforced.

As soon as the working copy is created, it should be edited to replace several template parameter values with customer-specific ones. For more detailed explanation, refer to Appendix A – LDAPACME$CONFIG-STD.INI_TEMPLATE, which contains a complete copy of the configuration file template.

Edit the working copy of the configuration file using the colored markup in Appendix A.

- The lines highlighted in yellow must be edited as follows:
  - All these lines should be ‘uncommented’ (i.e., remove the exclamation mark from the beginning of the line.

CAUTION! Leaving a line ‘commented’ can cause unpredictable consequences, including the inability to login to the system under any account.

  - Replace the lines’ contents with:
    - **server** — This should contain the IP address (or DNS Host name) of the authenticating server (domain controller)
      More than one server can be used (refer to the comments in the template configuration file).
    - **port** — This should retain the value “389” (the standard LDAP port).
    - **bind_dn** — This should be assigned the distinguished name of a domain user account that is used for the LDAP access. **This account should not have administrator privileges.**
    - **bind_password** — This should be assigned the password for the above user account.
    - **bind_timeout** — This should retain the value (*3*) from the template.
    - **base_dn** — This should be assigned the location in the directory directly beneath the authentication records.
    - **login_attribute** — This should be assigned the value “*samaccountname*” in place of “*uid*”.
    - **scope** — This value should not be changed; it must stay “*sub*”.
    - **port_security** — This should be set to “*none*”.
    - **password_type** — This must be assigned the value of “*active-directory*”.
    - **password_update** — The “*replace*” value should not be changed.

- The lines highlighted in cyan should be edited only in cases where there is at least one user whose name is different than the one in the Active Directory.
    - Both lines highlighted in cyan should be 'uncommented'.
        - **mapping** — This value should stay "*local*".
        - **mapping_file** — This name should be set to "LDAP_LOCALUSER_DATABASE.TXT" as described in Section 3 below.

3. Creating the LDAP Username Database

The username database file name can also be arbitrary. Like the above case (LDAP_LOCALUSER_DATABASE.TXT), although it has been preselected in order to simplify and automate the setup.

This step is only required if there are user(s) whose VMS username and AD username are different. However, from a system integrity standpoint (and to make the system ready to accommodate for this scenario), *the file should exist on a system even if it initially contains no information*.

Like in Section 2 above, you should start by locating the username database file template (SYS$COMMEN:[SYS$STARTUP]LDAP_LOCALUSER_DATABASE.TXT_TEMPLATE) and then copy it to the development location with the name: LDAP_LOCALUSER_DATABASE.TXT.

Edit the created username database file (if necessary) by adding the line(s) containing the comma-delineated username pairs. For example, if *User1* has the Windows name: "MyWindowsName", while also having the VMS name: "MyVMSName", then the corresponding added line should look like:

```
MyWindowsName,MyVMSName
```

NOTE: Usernames are not case sensitive.

If a name contains whitespaces, the corresponding entry in the username database must be quoted.

For example, if the Windows username is "My Windows Name" but the VMS name is like above, the added line should look like:

```
"My Windows Name",MyVMSName
```

An example of username database is shown in Appendix B – Username Database Example.

4. Creating an ACME Start File

The ACME Start file is: “SYS$MANAGER:ACME$START.COM”. It should be copied to your preselected development location and then edited.

1. Insert the following line so that it is the first executable line in the file:

```
$ DEFINE /SYSTEM /EXECUTIVE LDAPACME$INIT SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-
STD.INI
```

2. Locate and ‘uncomment’ the following line:

```
! @SYS$STARTUP:LDAPACME$STARTUP-STD             ! LDAP-STD
```

5. Creating the User Accounts Modification File

This file should be created from scratch (unless a previously created file will suffice). It is a command file that changes the preselected users’ accounts to require external authentication (AD).

The file should look like:

```
$ TOOLS
$ !SET VERIFY
$ IF P1 .EQS. ""
$ THEN
$ SAY := "!"
$ ELSE
$ SAY := "WRITE ''P1'"
$ ENDIF
$ L_COUNT == 0
$ !
$ DEFINE SYS$OUTPUT NL:
$ DEFINE SYS$ERROR NL:
$ CALL SUB_MOD <username 1> ON
.
.
.
$ CALL SUB_MOD <username M> ON
$ DEASSIGN SYS$ERROR
$ DEASSIGN SYS$OUTPUT
$ SAY "Modified " + F$STRING(L_COUNT) + " user accounts"
$ !SET NOVERIFY
$ DELETE /SYMBOL /GLOBAL L_COUNT
$ EXIT

$ SUB_MOD: SUBROUTINE
$ IF F$SEARCH("SYS$SYSDEVICE:[USERS]''P1'.DIR") .EQS. ""
$ THEN
$ SAY "User ''P1' does not exist"
$ ELSE
$ IF P2 .EQS. "ON"
$ THEN
$ AUTHORIZE MODIFY 'P1' /FLAGS=(EXTAUTH,VMSAUTH,PWDMIX)
$ ELSE
$ AUTHORIZE MODIFY 'P1' /FLAGS=(NOEXTAUTH,NOVMSAUTH,NOPWDMIX)
$ ENDIF
$ L_COUNT == L_COUNT + 1
$ ENDIF
$ EXIT
$ ENDSUBROUTINE
```

As shown in the template above, the file should have a line that calls a special subroutine for each username that will be affected (color-coded in the template). The file should be called "LDAP_ACCOUNTMODIFY.COM".

6. Disassociating User Accounts from External Authentication

The command file, "LDAP_ACCOUNTOFF.COM" performs the opposite function of the above "LDAP_ACCOUNTMODIFY.COM". It modifies all user accounts, prohibiting external (AD) user authentication. This file is used by the uninstallation procedure and looks like:

```
$ TOOLS
$ !SET VERIFY
$ IF P1 .EQS. ""
$ THEN
$ SAY := "!"
$ ELSE
$ SAY := "WRITE ''P1'"
$ ENDIF
$ !
$ DEFINE SYS$OUTPUT NL:
$ DEFINE SYS$ERROR NL
$ CALL SUB_MOD * OFF
$ DEASSIGN SYS$ERROR
$ DEASSIGN SYS$OUTPUT
$ SAY "Modified user accounts"
$ !SET NOVERIFY
$ DELETE /SYMBOL /GLOBAL L_COUN
$ EXIT

$ SUB_MOD: SUBROUTINE
$ IF F$SEARCH("SYS$SYSDEVICE:[USERS]''P1'.DIR") .EQS. ""
$ THEN
$ SAY "User ''P1' does not exist"
$ ELSE
$ IF P2 .EQS. "ON"
$ THEN
$ AUTHORIZE MODIFY 'P1' /FLAGS=(EXTAUTH,VMSAUTH,PWDMIX)
$ ELSE
$ AUTHORIZE MODIFY 'P1' /FLAGS=(NOEXTAUTH,NOVMSAUTH,NOPWDMIX)
$ ENDIF
$ ENDIF
$ EXIT
$ ENDSUBROUTINE
```

7. Modifying the VMS System Startup File

Copy “SYS$MANAGER:SYSTARTUP_VMS.COM” to the preselected development location and edit the copy by navigating to the last line of the file and adding:

```
$ SET SERVER ACME /RESTART
```

8. Modifying the ACME System Installation File

In order to automate the setup procedure, the command files cannot be interactive (they will be executed on all system nodes simultaneously and should not need the user’s input). The system file “SYS$MANAGER:SYS$LOGIN_SWITCH.COM” must be modified and updated. The modified file can be viewed in Appendix C – Modified SYS$ACM LOGIN and ACMELDAP (changes are color-coded).

9. Creating the Setup File

This file should be the same from one system to another; it can be created once and then become a permanent part of the MISER distribution. It should be named: “SYS$COMMON:[SYS$STARTUP]AD_NODE_SETUP.COM”. The contents should be:

```
$ IF F$SEARCH( "SYS$ERRORLOG:ADSETUP.LOG" ) .EQS. ""
$ THEN
$ OPEN LOGF SYS$ERRORLOG:ADSETUP.LOG /WRITE /ERROR=DOOR
$ ELSE
$ OPEN LOGF SYS$ERRORLOG:ADSETUP.LOG /APPEND /ERROR=DOOR
$ ENDIF
$ SAY := "WRITE LOGF "
$ SAY F$TIME()
$ OLDPRIV = F$SETPRV( "SYSPRV" )
$ IF .NOT. F$PRIV( "SYSPRV" )
$ THEN
$ SAY "*** You have insufficient access level to run this procedure ***"
$ GOTO DOOR
$ ENDIF
$ !
$ f_err == 0
$ ! Sanity check. Make sure all files (besides this file and its caller) are in place
$ CALL CHECKF SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI_TEMPLATE
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYS$STARTUP]LDAP_LOCALUSER_DATABASE.TXT
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYSMGR]ACME$START.COM_TEMPLATE
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYSMGR]ACME$START.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:LDAP_ACCOUNTMODIFY.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:SYSTARTUP_VMS.COM_TEMPLATE
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:SYSTARTUP_VMS.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYSMGR]SYS$LOGIN_SWITCH.COM_TEMPLATE
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$COMMON:[SYSMGR]SYS$LOGIN_SWITCH.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:AD_NEWUSER.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:AD_UNINSTALL.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$MANAGER:LDAP_ACCOUNTOFF.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ CALL CHECKF SYS$STARTUP:LDAPACME$STARTUP-STD.COM
$ IF f_err .NE. 0 THEN GOTO ERROR_EXIT
$ ! Protect sensitive files
$ SET SECURITY /PROTECTION = (system: "RWED", OWNER:"", GROUP:"", WORLD:"")
SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI
$ SET SECURITY /PROTECTION = (system: "RWED", OWNER:"", GROUP:"", WORLD:"")
SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT
$ SET SECURITY /PROTECTION = (system: "RWED", OWNER:"", GROUP:"", WORLD:"")
SYS$COMMON:[SYSMGR]SYS$LOGIN_SWITCH.COM
$ ! Install the SYS$ACM enabled kits
$ @SYS$MANAGER:SYS$LOGIN_SWITCH.COM ON LOGF
$ ! Setup the persona extension
$ ! ***************************
$ found = 0
$ def_idxfile = "SYS$UPDATE:VMS$SYSTEM_IMAGES.IDX"
$ IF F$SEARCH( "''def_idxfile'" ) .EQS. "" THEN GOTO POST_LOOP
$ idxfile = F$PARSE( P1, def_idxfile,,,"SYNTAX_ONLY")
$ OPEN /READ /ERROR = END_LOOP idxdat 'idxfile'
$ READ_LOOP:
$ READ /END = END_LOOP idxdat record
$ ldfile = F$EDIT( F$EXTRACT( 8, 39, record ), "TRIM,UPCASE" )
$ IF ldfile .EQS. "LDAPACME$EXT"
$ THEN
$ found = 1
$ GOTO END_LOOP
$ ELSE
$ GOTO READ_LOOP
$ ENDIF
$ END_LOOP:
```

```
$ CLOSE idxdat
$ POST_LOOP:
$ IF found .EQ. 0
$ THEN
$ MCR SYSMAN SYS_LOADABLE ADD LDAPACME LDAPACME$EXT
$ ENDIF
$ @SYS$UPDATE:VMS$SYSTEM_IMAGES
$ ! ***************************
$ ! Apply changes to the user accounts
$ @SYS$MANAGER:LDAP_ACCOUNTMODIFY.COM LOGF
$ !
$ SAY "AD Authentication setup completed successfully!"
$ GOTO DOOR
$ !
$ ERROR_EXIT: SAY "Setup failed"
$ GOTO DOOR
$ !
$ DOOR:
$ DELETE /SYMBOL /GLOBAL f_err
$ CLOSE LOGF
$ EXIT
$ !
$ CHECKF: SUBROUTINE
$ f_err == 0
$ IF F$SEARCH( "''P1'" ) .EQS. ""
$ THEN
$ SAY "''P1' cannot be located"
$ f_err == 1
$ ENDIF
$ EXIT
$ ENDSUBROUTINE
```

10. Creating the Initialization File

This file is used to distribute the necessary setup files to their proper locations on all system nodes and then execute the setup procedure afterwards. Like the setup file described earlier, this file does not depend on a particular system (it is generic) and therefore can be a permanent part of the MISER distribution. The file should be named “AD_SETUP.COM” and look like:

```
$ ! Save caller's location and verification state
$ CUR_DIR = F$DIRECTORY()
$ VER_ST = F$ENVIRONMENT("VERIFY_PROCEDURE")
$ SET NOVERIFY
$ !
$ TOOLS
$ OLDPRIV = F$SETPRV( "SYSPRV" )
$ IF .NOT. F$PRIV( "SYSPRV" )
$ THEN
$ WRITE SYS$OUTPUT "*** You have insufficient access level to run this procedure ***"
$ GOTO DOOR
$ ENDIF
$ !
$ CD = F$ENVIRONMENT("PROCEDURE")
$ DEV = F$PARSE(CD,,,"DEVICE")
$ FOLDER = F$PARSE(CD,,,"DIRECTORY")
$ FULLDIR = DEV+FOLDER
$ SET DEF 'FULLDIR'
$ !
$ CALL SUB_COPY LDAPACME$CONFIG-STD.INI_TEMPLATE SYS$COMMON:[SYS$STARTUP] 'P1'
$ CALL SUB_COPY LDAPACME$CONFIG-STD.INI SYS$COMMON:[SYS$STARTUP] 'P1'
$ CALL SUB_COPY LDAP_LOCALUSER_DATABASE.TXT SYS$COMMON:[SYS$STARTUP] 'P1'
$ CALL SUB_COPY ACME$START.COM_TEMPLATE SYS$COMMON:[SYSMGR] 'P1'
$ CALL SUB_COPY ACME$START.COM SYS$COMMON:[SYSMGR] 'P1'
$ CALL SUB_COPY LDAP_ACCOUNTMODIFY.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY SYSTARTUP_VMS.COM_TEMPLATE SYS$MANAGER: 'P1'
$ CALL SUB_COPY SYSTARTUP_VMS.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY SYS$LOGIN_SWITCH.COM_TEMPLATE SYS$COMMON:[SYSMGR] 'P1'
$ CALL SUB_COPY SYS$LOGIN_SWITCH.COM SYS$COMMON:[SYSMGR] 'P1'
$ CALL SUB_COPY AD_NODE_SETUP.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY AD_SETUP.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY AD_SETUP_LOCAL.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY AD_NEWUSER.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY AD_UNINSTALL.COM SYS$MANAGER: 'P1'
$ CALL SUB_COPY LDAP_ACCOUNTOFF.COM SYS$MANAGER: 'P1'
$ !
$ WRITE SYS$OUTPUT " "
$ WRITE SYS$OUTPUT "Setting up node(s)"
$ WRITE SYS$OUTPUT " "
$ ! The parameter accepted would be LOCAL - to run on the current node only
$ !
$ IF P1 .EQS. "LOCAL"
$ THEN
$  @SYS$MANAGER:AD_NODE_SETUP.COM
$ ELSE
$  NET_CMD @SYS$MANAGER:AD_NODE_SETUP.COM
$ ENDIF
$ WRITE SYS$OUTPUT "You have to reboot all nodes to complete setup"
$ DOOR:
$ ! Restore the verification state
$ SET NOVERIFY
$ IF VER_ST THEN SET VERIFY
$ SET DEF 'CUR_DIR'
$ EXIT
$ !
$ SUB_COPY: SUBROUTINE
$  WRITE SYS$OUTPUT "Copying ''P1' to ''P2'"
$  !DEFINE SYS$OUTPUT NL:
$  DEFINE SYS$ERROR NL:
$ ! Rename twice since the destination directory
$ ! may be specified by a logical referring to 2
$ ! physical directories (hopefully no more than that)
$ !
$  IF( P3 .EQS. "LOCAL" )
$ ! For local node
$  THEN
$   F = F$SEARCH( "''P2'''P1'" )
$   IF F .NES. ""
$   THEN
$    PURGE 'P2''P1'
$    REN 'P2''P1' 'P2''P1'_prev
$   ENDIF
$   F = F$SEARCH( "''P2'''P1'" )
$   IF F .NES. ""
$   THEN
$    REN 'P2''P1' 'P2''P1'_prev
$   ENDIF
$   COPY 'P1' 'P2'
$  ELSE
$ ! For all nodes
$   NET_CMD PURGE 'P2''P1'
$   NET_CMD REN 'P2''P1' 'P2''P1'_prev
$   NET_CMD REN 'P2''P1' 'P2''P1'_prev
$   COPY 'P1' 'P2'
$   NET_DIST 'P2''P1'
$  ENDIF
$  DEASSIGN SYS$ERROR
$  !DEASSIGN SYS$OUTPUT
$  EXIT
$ ENDSUBROUTINE
```

11. Creating the Local Initialization File

This file should only be used to install the AD authentication option on a single node. The files should be copied onto the destination node and then the command file “AD_SETUP_LOCAL.COM” should be executed. The file contents look like:

```
$ @AD_SETUP LOCAL
$ EXIT
```

12. Creating a New User Authentication Setup File

When a new user is added to the VMS **SYSUAF** database, it may be necessary to enforce the AD authentication. The command file implementing this task should look like:

```
$ SAY := "WRITE SYS$OUTPUT"
$ OLDPRIV = F$SETPRV( "SYSPRV" )
$ IF .NOT. F$PRIV( "SYSPRV" )
$ THEN
$ SAY "*** You have insufficient access level to run this procedure ***"
$ EXIT
$ ENDIF
$ USERNAME = P1
$ IF P2 .NES. ""  .AND.  P2 .NES. P1
$ THEN
$ ! AD and VMS usernames are different
$ USERNAME = P2
$ OPEN /APPEND /ERROR=ERR USERS SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT
$ WRITE USERS P1,",",P2
$ CLOSE USERS
$ NET_CMD PUR SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT
$ NET_DIST SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT
$ !
$ ENDIF
$ NET_CMD TOOLS
$ OPEN OF SYS$MANAGER:ADDUSER_TEMP.COM /WRITE /ERROR=ERR2
$ SAVE := "WRITE OF "
$ SAVE "$ DEFINE SYS$OUTPUT NL:"
$ SAVE "$ DEFINE SYS$ERROR NL:"
$ SAVE “$ TOOLS”
$ SAVE "$ AUTHORIZE MOD ''USERNAME' /FLAGS=(EXTAUTH,VMSAUTH,PWDMIX)"
$ SAVE "$ DEASSIGN SYS$ERROR"
$ SAVE "$ DEASSIGN SYS$OUTPUT"
$ CLOSE OF
$ NET_DIST SYS$MANAGER:ADDUSER_TEMP.COM
$ NET_CMD @SYS$MANAGER:ADDUSER_TEMP
$ NET_CMD DEL SYS$MANAGER:ADDUSER_TEMP.COM;*
$ SAY "USER ",USERNAME," AD authentication is set"
$ EXIT
$ ERR:
$ SAY "Could not open SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT"
$ EXIT
$ ERR2:
$ SAY "Could not open SYS$MANAGER:ADDUSER_TEMP.COM"
$ EXIT
```

13. Creating a Command File for Turning off AD Authentication

This command file (AD_UNINSTALL.COM) should have the following contents:

```
$ TOOLS
$ !SET VERIFY
$ SAY := "WRITE SYS$OUTPUT"
$ OLDPRIV = F$SETPRV( "SYSPRV" )
$ IF .NOT. F$PRIV( "SYSPRV" )
$ THEN
$ SAY "*** You have insufficient access level to run this procedure ***"
$ GOTO DOOR
$ ENDIF
$ IF P1 .NES. ""  .AND.  P1 .NES. "ALL"
$ THEN
$ SAY "Command file parameter - if specified - must be ALL"
$ GOTO DOOR
$ ENDIF
$ !
$ COPY SYS$COMMON:[SYSMGR]ACME$START.COM_TEMPLATE SYS$COMMON:[SYSMGR]ACME$START.COM
$ COPY SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI_TEMPLATE
SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI
$ COPY SYS$MANAGER:SYSTARTUP_VMS.COM_TEMPLATE SYS$MANAGER:SYSTARTUP_VMS.COM
$ IF P1 .EQS. "ALL"
$ THEN
$ NET_DIST SYS$COMMON:[SYSMGR]ACME$START.COM
$ NET_DIST SYS$COMMON:[SYS$STARTUP]LDAPACME$CONFIG-STD.INI
$ NET_DIST SYS$MANAGER:SYSTARTUP_VMS.COM
$ !
$ NET_CMD @SYS$COMMON:[SYSMGR]SYS$LOGIN_SWITCH.COM OFF
$ !
$ SAY "--- Restore users accounts"
$ NET_CMD @SYS$MANAGER:LDAP_ACCOUNTOFF.COM
$ !
$ SAY "AD Authentication was uninstalled!"
$ SAY "You have to reboot all nodes to restore VMS authentication"
$ ELSE
$ @SYS$MANAGER:SYS$LOGIN_SWITCH.COM OFF
$ AUTHORIZE MOD * /FLAGS=(NOEXTAUTH,NOVMSAUTH,NOPWDMIX)
$!
$ SAY "AD Authentication was uninstalled!"
$ SAY "You have to reboot this node to restore VMS authentication"
$ ENDIF
$ DOOR:
$ !SET NOVERIFY
$ EXIT
```

Appendix A LDAPACME$CONFIG-STD.INI_TEMPLATE

Below are the LDAP Initialization File contents. Refer to Section 2 – Creating the LDAP ACME Configuration File for information on how to treat highlighted text.

```
! Copyright 2013 Hewlett-Packard Development Company, L.P.
!
! This file is a template to help you create your own initialization
! file which will be read by the HP OpenVMS LDAP SYS$ACM Authentication Agent
! when it starts up.  It contains a set of information which determines
! how the agent should find the LDAP servers which contain authentication
! information.
!
! At start up time, the agent will use the logical name "LDAPACME$INIT"
! to find this file
!
! To create your own file, you can edit this file and replace the example
! parameters with information that corresponds with your own environment.

! Use the "server" directive to provide the IP address (or DNS host name)
! for your directory server.
!
! On OpenVMS V8.4 and above, you can specify one or more redundant servers
! by providing spaces between the server name/IP address.
!
! ex 1. server = test1.testdomain.com test2.testdomain.com
! ex 2. server = test1.testdomain.com test2.testdomain.com test3.testdomain.com
!
! The ACME LDAP tries to connect to first server first. If the connection fails
! for first server, the second server is tried for connection. If second server
! connection fails, the next set of server is tried in sequence, until the last
! server in the list. This applies to use search timeout as well.
!
! Note while using redundant servers:
!  1.) The base_dn, bind_dn and bind_password should be same on all the redundant
!      directory servers. The user records getting authenticated using ACME LDAP
!      should also be present on all the directory servers.
!
!  2.) Set the bind_timeout directive when using redundant multiple servers. This
!      ensures that the ACME LDAP tries to connect to all the redundant servers
!      before the user session times out.
!
!  3.) In case you have provided the Certificate Authority's (CA) public key
!      (ca_file directive) and the public keys are different, provide all the
!       public keys in the same ca_file. See comments around ca_file directive.
!

!server = 127.0.0.1

! Use the "port" tag to specify the LDAP port for connecting to the
! LDAP server. The default "port" is 389.
```

```
!port = 389

! Use the "bind_dn" tag to specify an authentication distinguished name (DN)
! in LDAP format which the agent will use when binding to each of the
! servers in your list.

!bind_dn = cn=admin,dc=hp,dc=com

! Use the "bind_password" tag to specify a password to go with the
! authentication DN.
!

!bind_password = adminpassword

! Use the "bind_timeout" directive, if you are providing multiple redundant
! servers in the "server" directive.
!
! Each bind request to a directory server, will by default take around 75
! seconds (TCPIP default connection establishment timeout), if the directory
! server is not reachable.
!
! If there are multiple redundant servers, the user login session (say a
! telnet session) will expire (within approximately 30 seconds), before
! ACME LDAP agent could check the list of all servers mentioned in the
! "server" directive.
!
! The bind_timeout takes a timeout value in seconds for connecting to one
! directory server in the list of all servers mentioned in the "server" directive.
! If you have say 2 servers mentioned in the .server. directive and bind_timeout
! is set to 3 seconds, the overall timeout period is around 6 seconds.

!bind_timeout = 3
!
! uncomment the following to use alternate server in case of search timeout while
! using redundant servers. The "server" directive can have more than one server
! mentioned as space (single) seperated list.
!
!search_timeout = 3
! Use the "base_dn" tag to specify the location in the directory underneath
! which the authentication records are kept:

!base_dn = dc=hp,dc=com

! Attribute to map from principal to LDAP entry

!login_attribute = uid

! Scope to search for an LDAP entry
!  sub: searches the base entry and all entries at all levels below the base entry
!  one: searches all entries at one level below the base entry
!  base: searches only the base entry

!scope = sub
```

```
! Filter for searching directory objects for valid user accounts (defaults
! to objectclass=*

!filter = objectclass=*

! Use the "port_security" directive to control how communications over the LDAP port
! are secured. The default is "starttls".
!
! The possible values for "port_security" are:
!
! starttls                           (negotiate SSL/TLS over standard LDAP port)
! ssl                                (this is an SSL-only port, e.g. port 636)
! none                               (no security - not recommended)

!port_security = starttls

! Password type for password changes
!  standard: use the standard userPassword attribute (default) on directory server
!  active-directory: use unicodePwd

!password_type = standard

! Password update method for changes to the standard password attribute (userPassword)
!  replace: use ldap_modify "replace" (default)
!  remove_and_add: use ldap_modify "remove-old/add-new"

!password_update = replace

! The LDAP SYS$ACM Authentication Agent will verify the validity of the LDAP
! server's public key certificate when using SSL. In order for this to happen
! you need to specify the location ("ca_file") of a file containing the Certificate
! Authority's (CA) public key used to sign the LDAP server's certificate.
!
! You can choose to disable this check by commenting out the line below.
!
! In case there are redundant servers having different public key certificate
! add the certificate information of the all the servers into the same file:
! example:
! $ type cacert.pem
! -----BEGIN CERTIFICATE-----
! .......
! server 1 public key certificate in base64 encoded format
! .......
! -----END CERTIFICATE-----
! -----BEGIN CERTIFICATE-----
! .......
! server 2 public key certificate in base64 encoded format
! .......
! -----END CERTIFICATE-----
! $
!
```

```
!ca_file = [directory]cacert.pem

!
! mapping for user name mapping whether global or local
!   Possible options are:
!
!   mapping is commented:
!           If mapping is commented, one-to-one mapping is used.
!           i.e. user name at "username: "prompt is the same as in sysuaf.dat file
!
!   server: (global mapping) Mapping between user name entered at "Username:" prompt
!           and the sysuaf.dat user account name happens based on some attributes
!           on the directory server.
!           You need to provide the mapping_attribute and mapping_target directive
!           if you use mapping=server
!
!   local: Mapping between user name entered at "Username:" prompt and the sysuaf.dat
!          user account name happens based on local CSV database file.
!          You need to provide the mapping_file directive if you use mapping=local

!mapping = local


! This directive is applicable only for global mapping.
! Specifies the attribute on directory server that is used for user mapping.
! For example:
! mapping_attribute can be referenced to the description attribute for the user
! in the directory server.
!
!    mapping_attribute=description
!
! You can also use any newly created attribute on the directory server
! for mapping. The attribute should be an IA5 multi-valued string.

!mapping_attribute = description

!
! This directive is applicable only for global mapping.
!
! The mapping_target is searched in the value of directory server's
! mapping_attribute field.
! For example: Let the LDAP INI file have:
!    mapping_attribute=description
!    mapping_target= VMSUsers.hp.com
! Let the description (attribute in Directory Server) be populated with:
!   VMSUsers.hp.com/jdoe
! The LDAP ACME agent then searches in VMSUsers.hp.com/jdoe, for a prefix of
! VMSUsers.hp.com/(with a forward slash (/) along with the mapping_target).
! The rest of the value that is, jdoe. is considered as the user name present in
! SYSUAF.DAT file. If a multi-valued string attribute is used, the
VMSUsers.hp.com/jdoe
! must be one of the array elements of the multi-valued string.
```

```
!mapping_target=VMSUsers.hp.com

!
! This directive is applicable only for local mapping.
!
! Specifies the complete path of the text database file to be searched for mapping
users.
! A template file is available inSYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT_TEMPLATE.
! This file includes the LDAP username and VMS usernames separated by a comma, where
LDAP
! username is the name of the user in the domain.
!
! For information on how to populate and load the contents of the database file, see
! SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT_TEMPLATE

!mapping_file=SYS$STARTUP:LDAP_LOCALUSER_DATABASE.TXT

!This directive is applicable only for Multi-Domain support
!
!Specifies the domain name of the ldap directory server against which the users has to
be authenticated.
!
!domain = testdomain1
```

Appendix B Username Database Example

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! This is the Ldap username text database file
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! The Login user name used in a corporate network might be different from the VMS user
name in SYSUAF.
! This file is used to store the mapping information, for the LDAP user name and the
SYSUAF username.
!
!
! Note:
! ** PLEASE CHANGE THE PROTECTION OF THIS FILE TO (S: RWED,O:,G:,W:)
!      ONCE THE LDAP NAMES AND SYSUAF NAMES ARE POPULATED.
!
!
!
! The file is automatically read into the LDAP ACME agent during
! -  startup/restartup of LDAP ACME agent (i.e. restart of ACME_SERVER process).
! - The SYS$SYSTEM:LDAP_LOAD_LOCALUSER_DATABASE.EXE is invoked as follows
!      $ load_localuser_db:=="$SYS$SYSTEM:LDAP_LOAD_LOCALUSER_DATABASE.EXE"
!      $  load_localuser_db <this file name with complete path>
! This file is read if the "MAPPING" parameter is set to "LOCAL" and the
"MAPPING_FILE"
! parameter is specified to point to this file with path, in the ACME LDAP INI
configuration file.
!
! - when using multi-domain feature, the domain name should be specified as an
arguement to the executable to load local user database
!   for that domain as an example given below.
!      $ load_localuser_db:=="$SYS$SYSTEM:LDAP_LOAD_LOCALUSER_DATABASE.EXE"
!      $ load_localuser_db <this file name with complete path> <argument referring to
domain>
!      ex:
!          $load_localuser_db
SYS$COMMON:[SYS$STARTUP]LDAP_LOCALUSER_DATABASE_AMERICAS.TXT testdomain1
!          $load_localuser_db SYS$COMMON:[SYS$STARTUP]LDAP_LOCALUSER_DATABASE_EMEA.TXT
testdomain2
!
! Way to populate the file:
! 1.) This file takes the LDAP user name and the SYSUAF user name as CSV records.
!      I.e. One entry LDAP Name and SYSUAF name should be present in one line
!           Separated by a comma.
!           Example:  ldap_username,VMS_username
!
! 2.) Comments should be started with exclamation. Anything after exclamation is
discarded.
!
! 3.) If you have special characters like spaces, or even commas or exclamation in
!      LDPA user name, provide it within quotes. Example. "test user, 1!"
!      In case you want quote itself in a user name, prepend with another quote.
!      Example: if user name is:sample"1, provide the user name as
"sample""1",vmsusername
```

```
!
! 4.) The numbers of users is restricted to 10000 currently.
!
! 5.) The length of each line can be a maximum upto 512.
!
! Provide the LDAPusername, SYSUAF user name below
!
!
"user 1",JohnDoe
User2, LisaDoe
```

Appendix C Modified SYS$ACM LOGIN and ACMELDAP

Below is the modified SYS$ACM LOGIN and ACMELDAP kits installation file: SYS$LOGIN_SWITCH. Refer to Section 2 – Creating the LDAP ACME Configuration File for information on how to treat highlighted text.

```
$! *************************************************************************
$! *                                                                       *
$! * VMS SOFTWARE, INC. CONFIDENTIAL. This software is confidential        *
$! * proprietary software licensed by VMS Software, Inc., and is not       *
$! * authorized to be used, duplicated or disclosed to anyone without      *
$! * the prior written permission of VMS Software, Inc.                    *
$! * Copyright 2015 VMS Software, Inc.                                     *
$! *                                                                       *
$! *************************************************************************
$!
$! X-3   WBF                          Burns Fisher             December 17, 2015
$!      Fix typo from fixing typos from adding code review comments
$!
$! X-2   WBF                          Burns Fisher             November 23, 2015
$!      Fix typos from adding code review comments
$!
$! X-1   WBF                          Burns Fisher             November 19, 2015
$!      Initial checkin
$!
$! This procedure switches from the traditional LOGINOUT.EXE (aka LOGIN_AUF),
$! which accesses the UAF file directly for authenticating users, to a version
$! of LOGINOUT that uses the SYS$ACME system service instead.
$!
$ ON WARNING THEN GOTO ERR_EXIT
$ SET ON
$ ON CONTROL_Y THEN EXIT
$!
$ say :=  "write sys$output"
$!
$! Init the symbol to say login_uaf is there, not login_acme.
$!
$ LOGIN98$PRESENT = 0
$!
$! Check if ACME LOGIN environment is setup,
$! if yes set logical LOGIN98$PRESENT
$!
$ SPECLOG = F$SEARCH("SYS$SPECIFIC:[SYSEXE]LOGINOUT.EXE")
$ SPECP0 = F$SEARCH("SYS$SPECIFIC:[SYSEXE]SETP0.EXE")
$ IF SPECLOG .NES. "" .OR. SPECP0 .NES. ""
$ THEN
$    say ""
$    say "Your system has a LOGINOUT.EXE or SETP0 image in
$    say "SYS$SPECIFIC:[SYSEXE].  With this configuration you
$    say "must switch the ACME and SETP0 images manually."
$    say "This procedure will now exit."
$    say ""
$    EXIT
$ ENDIF
$!
$ LOGCOM = F$SEARCH("SYS$COMMON:[SYSEXE]LOGINOUT.EXE")
$ P0COM = F$SEARCH("SYS$COMMON:[SYSEXE]SETP0.EXE")
$ IF LOGCOM .EQS. "" .OR. P0COM .EQS. ""
$ THEN
$    say ""
$    say "This procedure can't find SYS$SYSTEM:LOGINOUT.EXE,
$    say "SETP0.EXE, or both.  With this configuration you
$    say "must switch the ACME and SETP0 images manually."
$    say "This procedure will now exit."
$    say ""
$    EXIT
$ ENDIF
$ DEFINE/USER SYS$OUTPUT _NLA0:
$ ANALYZE/IMAGE/SELECT=(IDENTIFICATION=IMAGE) 'LOGCOM'
$ DEASSIGN/NOLOG SYS$OUTPUT
$ !show sym ANALYZE$IDENTIFICATION
$ IF F$LOCATE("LOGIN_ACME",ANALYZE$IDENTIFICATION) .NE.
F$LENGTH(ANALYZE$IDENTIFICATION)
$ THEN
$     LOGIN98$PRESENT = 1
$ ENDIF
$!
$ if p1 .NES. “OFF”  .AND.  p1 .NES. “ON”  .AND.  p1 .NES. “”
$ then
$     say “Incorrect command file parameter.”
$     say “If at all present, it can only be 0 or 1:”
$     say “OFF - for switching to UAF LOGIN,”
$     say “ON - for switching to ACME LOGIN.”
$     say "This procedure will now exit."
$     EXIT
$ endif
$ !
$ if login98$present .eq. 1
$ then
$     if p1 .eqs. “”
$     then
$         say "You are currently using ACME LOGIN."
$         say "This procedure will switch to using UAF LOGIN."
$     else
$ ! If we are already there, just exit
$         if p1 .eqs. “ON” then EXIT
$         goto do_it
$     endif
$ else
$     if p1 .eqs. “”
$     then
$         say "You are currently using UAF LOGIN.
$         say "This procedure will switch to using ACME LOGIN"
$     else
$!        If we are already there, just exit
$         if p1 .eqs. “OFF” then EXIT
$         goto do_it
$     endif
$ endif
$ goto ask
$ askagain:
$ say "You must answer YES or NO"
$ ask:
$ INQUIRE yesno "Do you want to continue? (YES or NO)"
$ yesno = "''F$EXTRACT(0,1,yesno)'"
$ if yesno .eqs. "N" then exit
$ if yesno .nes. "Y" then goto askagain
$!
$! Ok, now we are going to copy the correct images into SYS$COMMON:[SYSEXE],
$!
$ DO_IT: ON CONTROL_Y THEN GOTO ERR_EXIT
$ IF LOGIN98$PRESENT .EQ. 1
$ THEN
$! Here we have login98, aka ACME LOGIN.  Switch to traditional:
$!
$    COPY SYS$SYSTEM:LOGIN_UAF.EXE SYS$COMMON:[SYSEXE]LOGINOUT.EXE;
$!
$    COPY SYS$SYSTEM:SETP0_UAF.EXE SYS$COMMON:[SYSEXE]SETP0.EXE;
$ ELSE
$! Here we have traditional login, aka UAF LOGIN.  Switch to ACME:
$    COPY SYS$SYSTEM:LOGIN_ACME.EXE SYS$COMMON:[SYSEXE]LOGINOUT.EXE;
$!
$    COPY SYS$SYSTEM:SETP0_ACME.EXE SYS$COMMON:[SYSEXE]SETP0.EXE;
$ ENDIF
$!
$  DOINSTALL = "INSTALL"
$  DOINSTALL REPLACE LOGINOUT
$  DOINSTALL REPLACE SETP0
$
$  if p2 .eqs. ""
$  then
$  say "The replacement procedure is complete.  You must issue
$  say "the commands"
$  say ""
$  say "$INSTALL REPLACE LOGINOUT"
$  say "$INSTALL REPLACE SETP0"
$  say ""
$  say "on any other cluster members using a common system"
$  say "disk with ''F$GETSYI("NODENAME")'."
$  else
$  say "The login replacement procedure completed successfully"
$  endif
$! Everything is done. deassign sys$output and sys$error
$! it was assigned to null device previously and exit 1
$!
$ EXIT 1
$ ERR_EXIT:
$!
$! There was some problem.
```

```
$!
$  say "An unexpected error happened.  Please check SYS$SYSTEM:LOGINOUT.EXE"
$  say "and SYS$SYSTEM:SETP0.EXE and replace them manually if necessary"
$  EXIT 1
```