TOSHIBA
Leading Innovation

TOSHIBA
BIOS WMI Interface Guide

Version 2017, rev 1.5

Contents

List of Tables

1 TOSHIBA Legal, Regulatory and Safety

Copyright, Disclaimer and Trademarks

Copyright

First edition December 2016.

Disclaimer

This manual has been validated and reviewed for accuracy. The instructions and descriptions it contains are accurate for your computer at the time of this manual's production. However, succeeding computers' BIOS and manuals are subject to change without notice. TOSHIBA assumes no liability for damages incurred directly or indirectly from errors, omissions or discrepancies between the computers' BIOS and the manual.

Trademarks

Intel and Intel vPro are trademarks or registered trademarks of Intel Corporation.

Windows, Microsoft and Windows logo are registered trademarks of Microsoft Corporation.

Target groups

This document is intended for IT administrators, IT specialists and service engineers that need to develop solutions for changing or controlling the TOSHIBA BIOS settings through the Windows Management Instrumentation (WMI) interface. The manual guides you through the features of Toshiba BIOS and exemplifies the usage of the WMI interface with script samples.

A deeper understanding of BIOS, PCs, Networking, WMI and Visual Basic script language is a prerequisite to reading this manual.

General precautions for changing BIOS settings

Be careful when you change the BIOS settings. If certain BIOS settings are not correctly configured, it is possible that:

- Some features or devices may not function properly.
- Computer or system boot failure occurs, possibly resulting in loss of data.

The BIOS Setup Screen can be accessed by pressing the **F2** key when the TOSHIBA logo appears at boot time. If changes made to the BIOS result in system malfunction or undesired system performance, enter the BIOS again and press **F9** to load Setup Defaults, and then press **F10** to save and exit BIOS.

TOSHIBA Support

If you require any additional help using your computer or if you are having problems operating the computer, you may need to contact TOSHIBA for additional technical assistance.

Before you call

Some problems you experience may be related to software or the operating system, it is important to investigate other sources of assistance first. Before contacting TOSHIBA, try the following:

- Review troubleshooting sections in the documentation for software and peripheral devices.
- If a problem occurs when you are running software applications, consult the software documentation for troubleshooting suggestions. Call the software company's technical support for assistance.
- Consult the dealer you purchased your computer and/or software from.

Where to write

If you are still unable to solve the problem and suspect that it is hardware related, write to TOSHIBA at the location listed in the accompanying warranty booklet.

2 Toshiba BIOS and the WMI Interface

Overview

IT administrators try to find easy and quick solutions to manage the settings of the client computers' BIOS. The Toshiba WMI interface offers a simple way to access the BIOS.

The Toshiba WMI interface enables the administrator to read and write all BIOS settings, reset the BIOS to factory settings, set and change passwords and modify the boot order.

Windows Management Instrumentation (WMI)

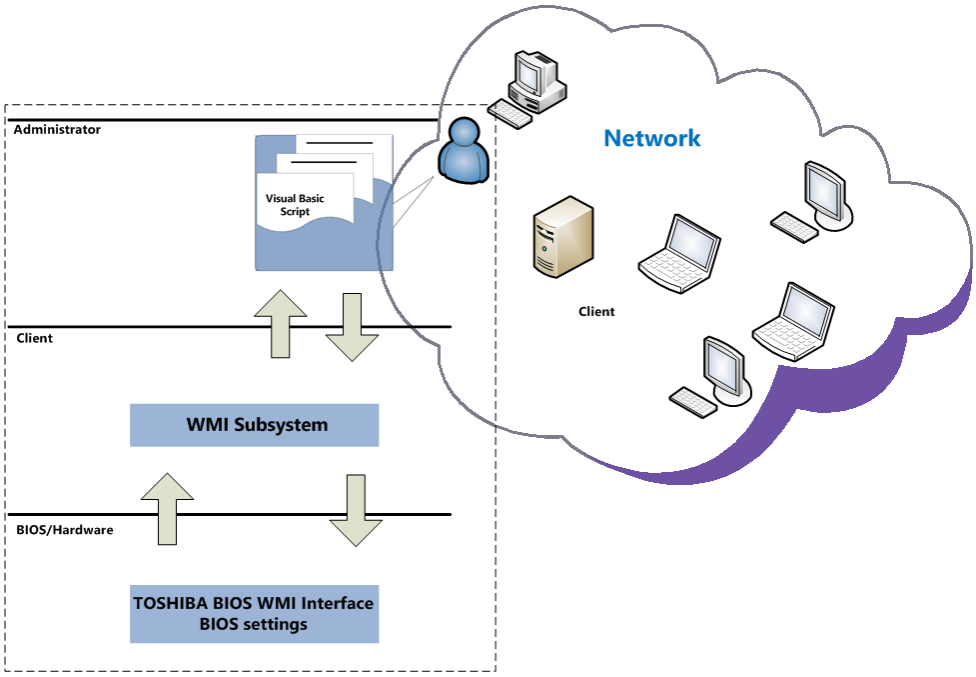
Windows Management Instrumentation (WMI) is in most Windows® operating systems included by default. It contains a wide range of functions:

- Start a process on a remote computer
- Schedule a process to be run on specific days at specific times
- Reboot a computer remotely
- Get a list of applications that are installed on a local computer or a remote computer
- Query the Windows event logs on a local computer or a remote computer

The Toshiba BIOS WMI provides additional functions. It facilitates the administration significantly.

The following illustration shows how the BIOS is controlled by WMI.

Figure 1 BIOS via WMI



Structure of the Toshiba BIOS

The Toshiba BIOS is divided into different areas – a general area with most BIOS settings intended for supervisors and a system management BIOS area (SMBIOS). As long as no supervisor password is set the general area and some fields of the SMBIOS area can be set freely. This is the default mode at the time the computer ships.

Once a supervisor password is set on the computer, which has to be done physically on each machine, the BIOS behavior changes. The BIOS will now require an authentication to modify any field values. Also a separate area of the BIOS will now be accessible. The supervisor will now have access to configure several security and boot related features.

The recommended scenario in a managed IT landscape is that the administrators set the supervisor password to restrict user access to the BIOS. This will prevent the users to modify any BIOS settings in an undesired way.

Passwords

The BIOS holds a number of passwords to restrict access either to the BIOS itself or to the hardware. Following is an explanation about each password and the area they protect. The passwords can partially be set, modified or deleted through the WMI interface either locally on a client or remotely over the network.

BIOS User Password

When the BIOS User Password is set, the user has to enter this password to access the BIOS through the Setup Utility. The Setup Utility can be accessed when pressing **F2** at boot time.

The following table shows a state diagram of the User Password's behavior.

Table 1 BIOS User Password

Current Status	Action	Arguments		
		1st	2nd (Old)	3rd (New)
Not Registered	Register	UserPassword	blank	PW
Registered	Change	UserPassword	PW	PW
Registered	Delete	UserPassword	PW	blank

For example: if a User Password is not set, it will be registered when calling **SetPassword** with "UserPassword" as first argument, a blank as second argument and a password as third argument. Once the User Password is registered, it can be either changed or deleted depending on the third argument in the call.

Examples on how to set the password through the WMI Interface will be given later in this chapter.

Note: *When setting, changing or deleting passwords though the WMI Interface, the password(s) delivered as arguments need to be encoded. The encoded password can be generated by accessing the following website: https://www.biospw.com/tsb/encoder/*

BIOS Supervisor Password

When the BIOS Supervisor Password is set, the BIOS will now require authentication to modify any BIOS setting. Access to several security configurations and boot related features will also now be available. These features include enable/disable certain built-in components as well as restrict specific boot devices.

Table 2 BIOS Supervisor Password

Current Status	Action	Arguments		
		1st	2nd (Old)	3rd (New)
Not Registered	N/A			
Registered	Change	SupervisorPassword	PW	PW
Registered	Delete	SupervisorPassword	PW	blank

The Supervisor Password cannot be set initially over the WMI Interface. It has to be set **manually** on the client first. It has to be set either manually on the client first or by separate toll that can only be executed locally. Once the Supervisor Password is set, it can be changed or deleted via WMI interface.

If you need a tool to locally set the Supervisor Password, please contact your Toshiba representative.

Note: *The password(s) delivered as arguments need to be encoded. The encoded password can be generated by accessing the following website: https://www.biospw.com/tsb/encoder/*

HDD Password

The HDD Password restricts the access to the HDD. There are two levels of passwords – a master password and a user password. The BIOS WMI Interface contains methods to set these passwords which are set physically onto the HDD.

When a HDD password is set, the HDD will require a valid password at boot time. When properly authenticated, the HDD can be accessed.

Note: *The HDD Password locks the HDD and is not stored in the BIOS. A locked HDD cannot be accessed even if it is installed in another computer. The contents on a common HDD are not encrypted when a HDD password is set. However, this may be different on Self Encrypting Drives (SED).*

Table 3 HDD Password

Current Status		Action	Arguments				
Master	User		1st	2nd (Old)	3rd (New)	4th (Old)	5th (New)
Not Registered	Not Registered	Register Master and User	Master+UserHDDPassword	blank	Master PW	blank	User PW
Registered	Not Registered	N/A					
Not Registered	Registered	N/A					
Registered	Registered	Change Only Master HDD password		Master PW	Master PW	blank	blank
		Change Only User HDD password		blank	blank	User PW	User PW
		Delete Master HDD password		Master PW	blank	blank	blank

		N/A					
Not Registered	Not Registered	N/A	UserOnlyHDDPassword				
Registered	Not Registered	N/A					
Not Registered	Registered	Change User HDD password		User PW	User PW		
Registered	Registered						

Note: *The password(s) delivered as arguments need to be encoded. The encoded password can be generated by accessing the following website: https://www.biospw.com/tsb/encoder/*

3 Using the TOSHIBA BIOS WMI Interface

This chapter contains details on the WMI implementation for configuring BIOS settings. The queries can be used to find settings and their values. The methods are used to set or change settings.

Configuring the BIOS settings

The following interface accesses the Toshiba BIOS settings.

Namespace: “\root\WMI”

Base Class: “ToshibaBIOSElement”

Queries

Table 4 Queries

Class Name	Type	Return types	Example
QueryTosIfVersion	Query	"Value"	"1.00"
QueryBiosSettings	Query	"Item,Attribute,Value	"WakeUpOnLAN,RW,Enabled"
QueryBiosSettingsParameter	Query	"Item,Value,…"	“WakeUpOnLAN,Disable,Enabl
QueryBiosItems	Query	”item”	“WakeOnLAN”
QuerySecurityPolicies	Query	"Item,Value"	"DeviceUSB,Enabled"
QuerySecurityPoliciesParamet	Query	"Item,Value,…"	“DeviceUSB,Disable,Enable”
QuerySecurityPolicyItems	Query	”item”	“DeviceUSB”
QueryPasswordStatus	Query	“Item,Value”	“UserPassword,Registered”
QueryPasswordItems	Query	“Item”	“UserPassword”
QuerySmbiosStrings	Query	"Item,Value"	"BoardAssetTag,DEFAULT"
QuerySmbiosItems	Query	”item”	“BoardAssetTag”

Note that all types returned are of type “String”. If there are several values returned, these will be separated by a comma (,)
The return type “Attribute” refers to the access rights of a specific BIOS Setting Item – RW (Read Write), RO (Read Only), WO (Write Only).

Items and Values are case sensitive strings. If you want to address a certain BIOS setting, the name and value need to be spelled correctly. It is recommended to use queries to find out the correct spelling and possible values for that setting.

Using the queries

Here is a VB example on using the QueryBiosSettings Query to find out all Bios Settings and their values.

```
'Sample VBScript: List all BIOS settings on the local computer
'
'command line: cscript.exe ListAllBiosSettings.vbs

On Error Resume Next
Dim objWMIService, objItem, colItems, strComputer, strSetting,
strItem, strValue

'define variables
strComputer = "."   'replace your computer name or leave "." as
default value

'connect to WMI
Set objWMIService = GetObject("winmgmts:\\" & strComputer &
"\root\WMI")

If Err.Number <> 0 Then
WScript.Echo "Unable to connect to WMI service: " &
Hex(Err.Number) & "."
WScript.Quit
End If

'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
QueryBiosSettings")

For Each objItem in colItems
If Len(objItem.CurrentSetting) > 0 Then
'return value contains elements, each separated by comma.
strSetting = ObjItem.CurrentSetting
strItem = Left(strSetting, InStr(strSetting, ",") - 1)
strValue = Mid(strSetting, InStr(strSetting, ",") + 1, 256)
WScript.Echo strItem + " = " + strValue
End If
Next

WScript.Quit
```

In a similar manner one can use the “QueryBiosSettingsParameter” query to find out possible parameters for each BIOS Setting:

```
Set colItems = objWMIService.ExecQuery("Select * from
QueryBiosSettingsParameter")
```

This is quite a useful query to understand the possible values for each setting. If you need only to find the name of a setting, the query "QueryBiosItems" will be the appropriate one.

Three queries are designed to access the Security Policy settings. Once a Supervisor Password is set physically on a computer, a supervisor can enable and disable hardware or boot media and set various security related parameter to restrict an ordinary user. If authenticated as supervisor during boot up, the restrictions can be bypassed. These queries can be used to read the current security policy settings.

QuerySecurityPolicyItems returns the names of the security policies, for instance "DeviceUSB"

QuerySecurityPolicies returns the security policies and settings

QuerySecurityPoliciesParameter returns the possible settings for each security policy

These two queries access the System management BIOS. The system management BIOS contain information about the manufacturer of the computer, the product name or serial number

QuerySmbiosItems returns the names of the items in SMBIOS

QuerySmbiosStrings returns the names and the values of the items in SMBIOS

There are two items in SMBIOS that can be written: *BoardAssetTag* and *ChassisAssetTagNumber*. These fields can be used for keeping track of hardware.

Each field can hold a maximum of 63 characters. If exceeded when setting, an error is returned.

Table 5 SMBIOS items with written property

SMBIOS item		Item name
Type2	Asset Tag	BoardAssetTag
Type3	Asset Tag Number	ChassisAssetTagNumber

The password queries return the names and status of the passwords. Please read the previous chapter on the different passwords and what they are used for.

QueryPasswordItems returns the names of the passwords

QueryPasswordStatus returns the name and the registration status of the password

The **QueryTosIfVersion** query returns the version of the Toshiba WMI Interface

Here is an overview of the queries and return values:

Table 6 Queries Overview

Use this query…		Return structure
QueryTosIfVersion		To get the version of Toshiba WMI interface
	Item	Version
QueryBiosSettings		To get the information of BIOS SETUP
	Item	The name of BIOS SETUP
	Attribute	The Read/Write attribute of BIOS SETUP
	Value	The value of the item

QueryBiosSettingsParameter		To get the acceptable settings of BIOS SETUP
	Item	The name of BIOS SETUP
	Value1, 2, 3, …	The value(s) of the item
QueryBiosItems		To get the item names of BIOS SETUP
	Item	The name of BIOS SETUP
QuerySecurityPolicies		To get the information of User Policy
	Item	The name of User Policy
	Value	The value of the item
QuerySecurityPoliciesParameter		To get the acceptable settings of User Policy
	Item	The name of User Policy
	Value1, 2, 3, …	The value(s) of the item
QuerySecurityPolicyItems		To get the item names of User Policy
	Item	The name of User Policy
QueryPasswordStatus		To get the registration status of passwords
	Item	The password name
	Value	The registration status of the password
QueryPasswordItems		To get the item names of passwords
	Item	The password name
QuerySmbiosStrings		To get the SMBIOS strings
	Item	The SMBIOS structure name of the string
	Value	The string
QuerySmbiosItems		To get the structure names of SMBIOS
	Item	The SMBIOS structure name of the string

Using a query on a remote computer

The following script reads the BIOS settings and their parameters on a remote computer.

```
'Sample VBScript: List all BIOS settings parameter on a remote computer
'
'command line: cscript.exe ListAllBiosSettingsParameterRemote.vbs
'[ComputerName|IPAddress] [UserName] [Password]
‘argument 1: the IP, the FQDN, or the Computer name of the client PC you want to access
'         2: a username that has Administrator privileges on the client PC you want to access
'         3: password for the username that has Administrator privileges on the client PC you want to access


On Error Resume Next
Dim objSWbemLocator, objWMIService, objItem, colItems, strComputer, strUserName, strPassword, strSetting, strItem, strValue


If WScript.Arguments.Count <> 3 Then
    WScript.Echo "Usage: cscript.exe ListAllBiosSettingsParameterRemote.vbs [ComputerName|IPAddress] [UserName] [Password]"
    WScript.Quit
```

```
End If

'define variables
strComputer = WScript.Arguments(0)  'computer name or computer's IP address
strUserName = WScript.Arguments(1)
strPassword = WScript.Arguments(2)

wbemImpersonationLevelImpersonate = 3
wbemAuthenticationLevelPktPrivacy = 6

'get the locator object
Set objSWbemLocator = CreateObject("WbemScripting.SWbemLocator")

'get the service object from the remote server
Set objWMIService = objSWbemLocator.ConnectServer(strComputer, "root\WMI",
strUserName, strPassword)

If Err.Number <> 0 Then
    WScript.Echo "Unable to connect to " & strComputer & ": " & Hex(Err.Number)
& "."
    WScript.Quit
End If

objWMIService.Security_.ImpersonationLevel = wbemImpersonationLevelImpersonate
objWMIService.Security_.AuthenticationLevel = wbemAuthenticationLevelPktPrivacy

'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
QueryBiosSettingsParameter")

For Each objItem in colItems
    If Len(objItem.CurrentSetting) > 0 Then
        'return value contains elements, each separated by comma.
        strSetting = ObjItem.ItemInformation
        If InStr(strSetting, ",") = 0 Then             'no comma found
            strItem = strSetting
            strValue = ""
        Else
            strItem = Left(strSetting, InStr(strSetting, ",") - 1)
            strValue = Mid(strSetting, InStr(strSetting, ",") + 1, 256)
        End If
        WScript.Echo strItem + " = " + strValue
    End If
Next

WScript.Quit
```

Methods

Methods are used to set and modify the BIOS settings. Here is an overview of the methods available.

Table 7 Method Overview

Class Name / Methods		Type	Instance name	Return/Parameter	Example
ModeControl					
	SetConfigurationMode	Method	ACPI\PNP0C14\0_0	"mode,svpw(*); "	"Start,1E302E; "
BiosSetting					
	GetBiosSetting	Method	ACPI\PNP0C14\0_0	"Item;"	"WakeUpOnLAN,Enabled"
	LoadDefaultBiosSettings	Method	ACPI\PNP0C14\0_1	"Execute;"	"Execute;"
	SetBiosSetting	Method	ACPI\PNP0C14\0_2	"Item,Value;"	"WakeUpOnLAN,Disable;"
SecurityPolicy					
	GetSecurityPolicy	Method	ACPI\PNP0C14\0_0	"Item;"	"DeviceUSB,Enabled"
	SetSecurityPolicy	Method	ACPI\PNP0C14\0_1	“Item,Value;”	“DeviceUSB,Disable;”
Password					
	GetPasswordStatus	Method	ACPI\PNP0C14\0_0	“Item;”	“UserPassword,Registered”
	SetPassword	Method	ACPI\PNP0C14\0_1	“Item,OldPwd(*), NewPwd(*)	“UserPassword,1E302E, 2D152C”
SmBiosString					
	GetSmbiosString	Method	ACPI\PNP0C14\0_0	"Item;"	"BoardAssetTag,12345678"
	SetSmbiosString	Method	ACPI\PNP0C14\0_1	"Item,Value;"	"BoardAssetTag,12345678;"

(*) The password(s) delivered as arguments need to be encoded. The encoded password can be generated by accessing the following website: https://www.biospw.com/tsb/encoder/

In the following table you will find each Method with a short explanation and the parameter they require.

Table 8 Method Details

Method		Parameter
SetConfigurationMode		If the supervisor password is registered, one needs to unlock the BIOS settings by sending Start and encoded Password. Start authenticates the supervisor. End closes the authentication.
	Mode	Start: Enter the mode where each BIOS Setting can be written when the supervisor password is registered End: Leave the mode where each BIOS Setting can be written when the supervisor password is registered
	svpw	Encoded supervisor password
GetBiosSetting		To get information about the BIOS Setting
	Item	The name of BIOS Setting
LoadDefaultBiosSettings		Loads the default settings of the BIOS SETUP
SetBiosSetting		To set a specific BIOS Setting
	Item	The name of BIOS Setting
	Value	The value of the item
GetSecurityPolicy		To get information of the User Policy
	Item	The name of the User Policy
SetSecurityPolicy		To set the User Policy
	Item	The name of the User Policy

Method		Parameter
	Value	The value of the item
GetPasswordStatus		To get the registration status of the password
	Item	The password name
SetPassword		To set a password
	Item	The name of passwords
	OldPwd	Old encoded password
	NewPwd	New encoded password
GetSmbiosString		To get the string of the SMBIOS
	Item	The SMBIOS structure name
SetSmbiosString		To set the strings of SMBIOS
	Item	The SMBIOS structure name of the string
	Value	The string

How to set a BIOS setting on a remote computer

The following code is a sample script on how to set a Bios setting on a remote computer. The script uses several methods and gives a representative example on how to use these methods. The scenario assumes that a supervisor password has been set (can only be set manually) on the remote computer.

You need an account with administrative rights on the remote computer to connect to. Otherwise you will not have enough privileges to modify a BIOS setting remotely.

Please save the following short script with the functions IsSupervisorPasswordRegistered and SetConfigurationMode into a file called “procedures.vbs”. The main script will load these functions into memory and use them when required.

```
'
'    Function   : Check if the supervisor password exists
'    parameter  : WMI service object
'
Function IsSupervisorPasswordRegistered(objWMIService)

    Dim isRegistered
    isRegistered = -1

    ''query the password status to check if the supervisor
password is registered
    Set colItems = objWMIService.ExecQuery("Select * from
Password where InstanceName='ACPI\\PNP0C14\\0_0'")

    For Each objItem in colItems
        'execute the method and obtain the return status
        objItem.GetPasswordStatus "SupervisorPassword;",
strReturn

        ''return value contains two elements, each seperated by
comma. e.g: "SupervisorPassword,Registered"
        strItem = Left(strReturn, InStr(strReturn, ",") - 1)
        strStatus = Mid(strReturn, InStr(strReturn, ",") + 1,
256)
```

```
        If strStatus = "Registered" Then
            isRegistered = 0
        End If
    Next

    IsSupervisorPasswordRegistered = isRegistered

End Function

'
'    Function   : authenticate/deauthenticate with Supervisor
privilege
'   parameter 1 : WMI service object
'             2 : input parameter value for mode control method
'
Function SetConfigurationMode(objWMIService, strInParamValue)
    Dim colItems, objItem

    'executes a WQL query
    Set colItems = objWMIService.ExecQuery("Select * from
ModeControl where InstanceName='ACPI\\PNP0C14\\0_0'")

    For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetConfigurationMode strInParamValue, strReturn
    Next

    SetConfigurationMode = strReturn

End Function
```

This is the main script on how set a single BIOS Setting on a remote computer.

```
'
'Sample VBScript: Set a single BIOS setting on a remote
computer. Use this script if you have registered a supervisor
password.
'
'command line: cscript.exe SetBiosConfigPasswordRemote.vbs
[setting] [value] [scrambled SupervisorPassword]
[ComputerName|IPAddress] [UserName] [Password]
'   argument  1 : BIOS item name
'             2 : the setting value you want to change
'             3 : the scrambled supervisor Password
'             4 : the IP, the FQDN, or the Computer name of the
client PC you want to access
'             5 : a username that has Administrator privileges
on the client PC you want to access
'             6 : password for the username that has
Administrator privileges on the client PC you want to access

'declare application name
Dim strAppName
strAppName = "SetBiosConfigPasswordRemote"

On Error Resume Next
Dim objFSO, objFile, objWMIService, objItem, colItems,
strComputer, strInParamValue, strReturn, strItem, strStatus,
```

```
strFileName, strSupervisorPassword, strParameter, strUserName,
strPassword

'create Object to open the procedure file
strFileName = "procedures.vbs"
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.OpenTextFile(strFileName, 1)   '1 - for
reading
Execute objFile.ReadAll()

''check input parameters
If WScript.Arguments.Count <> 6 Then
    WScript.Echo "Usage: cscript.exe
SetBiosConfigPasswordRemote.vbs [setting] [value] [scrambled
SupervisorPassword] [ComputerName|IPAddress] [UserName]
[Password]"
    WScript.Quit
End If

'define variables
strInParamValue = WScript.Arguments(0) + "," +
WScript.Arguments(1) + ";"
strSupervisorPassword = WScript.Arguments(2)
strComputer = WScript.Arguments(3)  'computer name or
computer's IP address
strUserName = WScript.Arguments(4)
strPassword = WScript.Arguments(5)

wbemImpersonationLevelImpersonate = 3
wbemAuthenticationLevelPktPrivacy = 6

'get the locator object
Set objSWbemLocator = CreateObject("WbemScripting.SWbemLocator")

'get the service object from the remote server
Set objWMIService = objSWbemLocator.ConnectServer(strComputer,
"root\WMI", strUserName, strPassword)

If Err.Number <> 0 Then
    WScript.Echo "Unable to connect to " & strComputer & ": " &
Hex(Err.Number) & "."
    WScript.Quit
End If

objWMIService.Security_.ImpersonationLevel =
wbemImpersonationLevelImpersonate
objWMIService.Security_.AuthenticationLevel =
wbemAuthenticationLevelPktPrivacy


''check if the supervisor password is registered
strReturn = IsSupervisorPasswordRegistered(objWMIService)
If strReturn <> 0 Then
    WScript.Echo "You can not run this application if the
supervisor password is not registered."
    WScript.Quit
End If


''authenticate with Supervisor privilege
```

```
strParameter = "Start," + strSupervisorPassword + ";"
strReturn = SetConfigurationMode(objWMIService, strParameter)
If strReturn <> 0 Then
    WScript.Echo "Supervisor password authentication failed.
Error: " & GetErrMsg(Hex(strReturn))
    WScript.Quit
Else
    WScript.Echo "Supervisor password successfully
authenticated."
End If

'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
BiosSetting where InstanceName='ACPI\\PNP0C14\\0_0'")

''set single Bios setting
For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetBiosSetting strInParamValue, strReturn
Next

WScript.Echo strAppName & ": " & GetErrMsg(Hex(strReturn))


''deauthenticate from supervisor mode
strParameter = "End," + strSupervisorPassword + ";"
strReturn = SetConfigurationMode(objWMIService, strParameter)
If strReturn <> 0 Then
    WScript.Echo "Supervisor password deauthentication failed.
Error: " & GetErrMsg(Hex(strReturn))
    WScript.Quit
Else
    WScript.Echo "Supervisor password successfully
deauthenticated."
End If

WScript.Quit


''convert an error code to a string
Function GetErrMsg(err)
    Dim strMsg
    Select Case err
        Case "0"
           strMsg = "The operation was successful."
        Case "8004100C"
           strMsg = "Feature or operation is not supported."
        Case "80041008"
           strMsg = "One of the parameters to the call is not
correct."
        Case "80041003"
           strMsg = "Write Protect error"
        Case "80041062"
           strMsg = "Operation failed because the client did
not have the necessary security privilege."
        Case "80045001"
           strMsg = "Authentication failure."
        Case "80045002"
           strMsg = "Password not registered."
        Case Else
```

```
            strMsg = "error code " + err
    End Select

    GetErrMsg = strMsg
End Function
```

The scripts explained

In this section, we will explain the key points of the scripts and how the WMI Methods and Queries are used. This part of the main script:

```
'create Object to open the procedure file
strFileName = "procedures.vbs"
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.OpenTextFile(strFileName, 1)   '1 - for
reading
Execute objFile.ReadAll()
```

will load the “procedures.vbs” file containing the additional functions into memory so that they can be called upon. This is a Visual Basic syntax for dynamically loading libraries.

The main script calls the `IsSupervisorPasswordRegistered` function is called to check if the supervisor password is set.

```
strReturn = IsSupervisorPasswordRegistered(objWMIService)
```

This calls the function in the “procedures.vbs” and the function will read whole password structure into a variable. By using the `GetPasswordStatus` method, the SupervisorPassword is filtered and its status is read.

```
Set colItems = objWMIService.ExecQuery("Select * from Password
where InstanceName='ACPI\\PNP0C14\\0_0'")

For Each objItem in colItems
   'execute the method and obtain the return status
    objItem.GetPasswordStatus "SupervisorPassword;", strReturn
   'return value contains two elements, each seperated by comma.
e.g:              "SupervisorPassword,Registered"
```

Now when the supervisor password is set on the remote computer, the main script needs to authenticate to be able to perform any changes to the remote BIOS.

The script calls `SetConfigurationMode` in the “procedures.vbs” file. The Parameters needed are Start, + the encoded Supervisor Password + `";"`.

The encoded password can be generated by accessing the following website: https://www.biospw.com/tsb/encoder/

```
''authenticate with Supervisor privilege
strParameter = "Start," + strSupervisorPassword + ";"
strReturn = SetConfigurationMode(objWMIService, strParameter)
```

The `SetConfigurationMode` function gets a handle on the *ModeControl* structure and uses the method *SetConfigurationMode* to “Start” the authentication.

```
Set colItems = objWMIService.ExecQuery("Select * from
ModeControl where InstanceName='ACPI\\PNP0C14\\0_0'")

For Each objItem in colItems
'execute the method and obtain the return status
objItem.SetConfigurationMode strParameter, strReturn
…
```

Now the main script is authenticated and can perform changes to the remote BIOS. Get a handle on the *BiosSettings* structure and use the *SetBiosSettings* Method to change the BIOS.

```
Set colItems = objWMIService.ExecQuery("Select * from
BiosSetting where InstanceName='ACPI\\PNP0C14\\0_0'")

''set single Bios setting
For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetBiosSetting strInParamValue, strReturn
Next
```

When exiting the script - do not forget to end the authentication of the remote supervisor password:

```
''deauthenticate from supervisor mode
strParameter = "End," + strSupervisorPassword + ";"
strReturn = SetConfigurationMode(objWMIService, strParameter)
```

Return values

You will receive one of the following return values after calling the WMI methods. Zero is returned on a successful operation. Other values are returned when an error occurs. The following return values are used by the Toshiba WMI interface. Please see the Microsoft Developer Network (MSDN) for other return values.

Table 9 Method Return Value

Return Value	Description
0 (0x0) WBEM_S_NO_ERROR	The operation was successful.
2147749900 (0x8004100C) WBEM_E_NOT_SUPPORTED	Feature or operation is not supported.
2147749896 (0x80041008) WBEM_E_INVALID_PARAMETER	One of the parameters to the call is not correct The system supports the subfunction but the subfunction is called with the invalid input arguments.
2147749891 (0x80041003) WBEM_E_ACCESS_DENIED	Write Protect error Examples: To enable CMP (Core Multi Processing) while TxT (Trusted eXecution Technology) is enabled. To change VT (Virtualization Technology) setting while TxT is enabled To enable TxT while either VT or CMP is disabled To change Boot Menu mode while it is restricted by the policy
2147749986 (0x80041062) WBEM_E_PRIVILEGE_NOT_HELD	Operation failed because the client did not have the necessary security privilege. Examples: To change BIOS settings if the supervisor password is registered but not provided by ConfigurationMode class
2147766273 (0x80045001) (Toshiba original)	Authentication failure Examples: An incorrect password is provided with ConfigurationMode class. An incorrect password is provided as the old password to change a password
2147766274 (0x80045002) (Toshiba original)	Password not registered Examples: ConfigurationMode is called but Supervisor password is not registered.

Other methods

Load Default Bios Settings

Most BIOS settings have a default value. There is a method to reset the BIOS settings with a predefined default value back to original state. This can be done with the method: LoadDefaultBiosSettings.

Excerpts out of a VB Script:

```
'connect to WMI
Set objWMIService = GetObject("winmgmts:\\" & strComputer &
"\root\WMI")

'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
BiosSetting where InstanceName='ACPI\\PNP0C14\\0_0'")

For Each objItem in colItems
    'execute the method and obtain the return status
    ObjItem.LoadDefaultBiosSettings "Execute;", strReturn
Next
```

System BIOS Settings

There are two fields in the System BIOS that can be set to store customer asset tag information.

Table 10 Asset Tag BIOS item

SMBIOS item		Item name
Type2	Asset Tag	BoardAssetTag
Type3	Asset Tag Number	ChassisAssetTagNumber

This script shows how to set any of these two strings in the SMBIOS.

```
'
'Sample VBScript: Set a single smbios string on the local
computer.  Use this script if you have no supervisor password
set.
'
'   command line: cscript.exe SetSmbiosConfig.vbs [string]
[value]
'   argument  1 : Smbios item name
'             2 : the string value you want to change

'declare application name
Dim strAppName
strAppName = "SetSmbiosConfig"

On Error Resume Next
Dim objWMIService, objItem, colItems, strComputer,
strInParamValue, strReturn, strItem, strStatus, strFileName,
strParameter

''check input parameters
If WScript.Arguments.Count <> 2 Then
    WScript.Echo "Usage: cscript.exe SetSmbiosConfig.vbs
[string] [value]"
    WScript.Quit
End If
```

```
'check write permissions
If WScript.Arguments(0) <> "BoardAssetTag" And
WScript.Arguments(0) <> "ChassisAssetTagNumber" Then
    WScript.Echo "You do not have permission to modify smbios
strings except BoardAssetTag and ChassisAssetTagNumber."
    WScript.Quit
End If


'define variables
strInParamValue = WScript.Arguments(0) + "," +
WScript.Arguments(1) + ";"
strComputer = "."   'replace your computer name or leave "." as
default value

'connect to WMI
Set objWMIService = GetObject("winmgmts:\\" & strComputer &
"\root\WMI")

If Err.Number <> 0 Then
    WScript.Echo "Unable to connect to WMI service: " &
Hex(Err.Number) & "."
    WScript.Quit
End If


'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
SmbiosString where InstanceName='ACPI\\PNP0C14\\0_0'")

''set single smbios string
For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetSmbiosString strInParamValue, strReturn
Next

WScript.Echo strAppName & ": " & GetErrMsg(Hex(strReturn))

WScript.Quit



''convert an error code to a string
Function GetErrMsg(err)
    Dim strMsg
    Select Case err
        Case "0"
           strMsg = "The operation was successful."
        Case "8004100C"
           strMsg = "Feature or operation is not supported."
        Case "80041008"
           strMsg = "One of the parameters to the call is not
correct."
        Case "80041003"
           strMsg = "Write Protect error"
        Case "80041062"
           strMsg = "Operation failed because the client did
not have the necessary security privilege."
        Case "80045001"
```

```
            strMsg = "Authentication failure."
        Case "80045002"
            strMsg = "Password not registered."
        Case Else
            strMsg = "error code " + err
    End Select

    GetErrMsg = strMsg
End Function
```

4 BIOS Settings

All the BIOS Settings are listed in the following table. The first two columns show the names of the BIOS setting and on which page they appear in the BIOS Setup Screen. The BIOS Setup Screen can be accessed by pressing **F2** during boot time.

Note that not all settings may be available. Many of the settings are dependent on the available hardware. To find out which settings are available please use the QueryBiosSettings query as described in Chapter 3.

Each of BIOS settings have their own WMI interface name, which is listed in the third column. The fourth column shows the access attribute of each item. The “Acceptable values” column lists all values that each item can take. It is recommended to use the QueryBiosSettingsParameter query to list each acceptable value.

The last column shows if a BIOS setting has a default value. The settings marked with Read Only (RO) cannot be set at all. The settings marked with Write Only (WO) can only be written. The settings marked with Y (Yes) has a default value whereas the settings with N (No) will keep the current setting even load default is executed. You can change the BIOS Settings to default using the LoadDefaultBiosSettings method as described in chapter 3.

Table 11 BIOS Settings

Page in SETUP	Item name in SETUP	WMI Item name	Attr.	Acceptable values	Default
Main	System BIOS Version	SystemBiosVersion	RO	-	-
	EC Version	ECVersion	RO	-	-
	AMT Setup Prompt	AMTSetupPrompt	RW	"Enable", "Disable"	Y
	Language	Language	RW	"English", "French"	N
Security	Secure Boot	SecureBoot (*6)	RW	"Enable"	N
	Clear Fingerprint data	ClearFingerprintdata	WO	"Execute"	-
	TPM	TPM (*1)	RW	"Enable", "Disable"	N
	Clear TPM Owner	ClearTPMOwner	WO	"Execute"	-
	Hide TPM	HideTPM	RW	"Yes", "No"	N
	Boot Menu	BootMenu	RW	"Enable", "Disable"	Y
	USB Provisioning of AMT	UsbProvisioningOfAmt	RW	"Enable", "Disable"	Y
Power Management	Wake-up on LAN	WakeUpOnLAN	RW	"Enable", "Disable"	Y
	Wake-up on LAN on Battery	WakeUpOnLANOnBattery	RW	"Enable", "Disable"	Y
	Wake on Keyboard	WakeOnKeyboard	RW	"Enable", "Disable"	Y
	Critical Battery Wake-up	CriticalBatteryWakeUp	RW	"Enable", "Disable"	Y
	Panel Open - Power On	PanelOpenPowerOn	RW	"Disable", "EnableSleepOnly", "EnableSleepAndOff"	Y
	Power on by AC	PowerOnByAc	RW	"Enable", "Disable"	Y
	Dynamic CPU Frequency Mode	DynamicCPUFrequencyMode	RW	"DynamicallySwitchable", "AlwaysHigh", "AlwaysLow"	Y
	Core Multi-Processing	CoreMultiProcessing	RW	"Enable", "Disable"	Y

	Intel Turbo Boost Technology	IntelTurboBoostsTechnology	RW	"Enable", "Disable"	Y
	Intel Display Power Management	IntelDisplayPowerManagement	RW	"Enable", "Disable"	Y
	eSATA	eSATA	RW	"Enable", "Disable"	Y
	SATA Interface setting	SATAInterfaceSetting	RW	"Performance", "BatteryLife"	Y
	Intel(R) Rapid Start Technology	IntelRapidStartTechnology	RW	"Enable", "Disable"	Y
	Rapid Start Entry after	RapidStartEntryAfter	RW	"Immediately", "10minutes", "2hours", "5hours", "24hours"	Y
	Internal USB3.0 Controller	InternalUSB30Controller	RW	"Enable", "Disable"	Y
	Keyboard Backlight Control Mode	KeyboardBacklightControlMode	RW	"TIMER", "ON", "OFF"	Y
	Backlight Lighting Time	BacklightLightingTime	RW	"00" - "60"	Y
Power Management (BIOS Power Management)	Battery Save Mode	BatterySaveMode	RW	"UserSetting", "LowPower", "FullPower"	Y
	Processing Speed	ProcessingSpeed	RW	"Low", "High"	Y
	CPU Sleep Mode	CPUSleepMode	RW	"Enable", "Disable"	Y
	LCD Brightness	LCDBrightness	RW	"Bright,SemiBright", "SuperBright"	Y
	Cooling Method	CoolingMethod	RW	"HighPerformance", "Balanced", "PowerSaver"	Y
	PCI Express Link ASPM	PCIExpressLinkASPM	RW	"Disable", "Enable", "Auto"	Y
Advanced	Execute-Disable Bit Capability	ExecuteDisableBitCapability	RW	"NotAvailable", "Available"	Y
	Virtualization Technology	VirtualizationTechnology (*1)	RW	"Disable", "VT-xAndVT-d", "VT-xOnly", "VT-dOnly"	N
	Trusted Execution Technology	TrustedExecutionTechnology (*2)	RW	"Enable", "Disable"	N
	SW Guard Extentions(SGX)	SWGuardExtensions	RW	"Disable","Enable","SoftwareControl"	N
	Intel(R) AT	IntelAT	RW	"Enable", "Disable"	Y
	Intel(R) AT Suspend	IntelATSuspend	RW	"Enable", "Disable"	Y
	Beep Sound	BeepSound	RW	"OFF", "Low", "Medium", "High"	Y
	Sleep and Charge	SleepAndCharge	RW	"Disable", "AutoMode", "20AMode"	Y
	System ON CDP Charge Mode	SystemOnCDPChargeMode	RW	"Enable", "Disable"	Y
	USB Power in Sleep Mode	USBPowerInSleepMode	RW	"Enable", "Disable"	Y
	USB Power in Off State	USBPowerInOffState	RW	"Enable", "Disable"	Y
	Sleep and Music	USBSleepAndMusic	RW	"Enable", "Disable"	Y
	USB Legacy Emulation	USBLegacyEmulation	RW	"ColdBootOnly", "Always"	Y
	USB Memory BIOS Support Type	USBMemoryBIOSSupportType	RW	"HDD", "FDD"	Y

	Change Boot Order	ChangeBootOrder (*3)	RW	"HDD/SSD", "USBMemory", "eSATAHDD", "USBODD", "FDD", "LAN", "ODD", "HDD2/SSD2"	Y
Advanced (System Configuration)	Built-in LAN	BuiltInLAN	RW	"Enable", "Disable"	Y
	Wireless LAN	WirelessLAN	RW	"Enable", "Disable"	Y
	Auto Wireless LAN RF Switching	AutoWirelessLANRFSwitching	RW	"Enable", "Disable"	Y
	Wireless WAN	WirelessWAN	RW	"Enable", "Disable"	Y
	WiMAX	WiMAX	RW	"Enable", "Disable"	Y
	Wireless Gigabit (WiGig)	WirelessGigabit	RW	"Enable", "Disable"	Y
	Bluetooth	Bluetooth	RW	"Enable", "Disable"	Y
	Internal Pointing Device	InternalPointingDevice	RW	"Enable", "Disable"	Y
	Web Camera	WebCamera	RW	"Enable", "Disable"	Y
	SD Host Controller	SDHostController	RW	"Enable", "Disable"	Y
	Fingerprint Sensor	FingerprintSensor	RW	"Enable", "Disable"	Y
	Internal Thunderbolt Controller	InternalTunderboltController	RW	"Enable", "Disable"	Y
	Memory Performance Mode	MemoryPerformanceMode	RW	"Enable", "Disable"	Y
	SATA Controller Mode	SATAControllerMode	RW	"IDEMode", "OSAHCIMode", "BIOSAHCIMode"	N
	LAN Boot Selection	LanBootSelection	RW	"BuiltInLAN", "ThunderboltDockLAN", "TypeCLAN"	Y
	Boot Mode	BootMode	RW	"CSMBoot", "UEFIBoot", "UEFIBoot-Legacy"	N
	Power On Display	PowerOnDisplay	RW	"Auto-Selected", "SystemLCDOnly", "LCD+ExternalDisplay"	Y
	Boot Up NumLock Status	BootUpNumlockStatus	RW	"ON", "OFF"	Y
	External Display Device	ExtendedBootDisplayDevice	RW	"AnalogRGB", "DisplayPort", "HDMI", "DVI"	Y
	Wait for monitor detection	WaitForMonitorDetection	RW	"Enable, "Disable"	Y
	Function Button	FunctionButton	RW	"Enable", "Disable"	Y
	Function Button Beep	FunctionButtonBeep	RW	"OFF", "Low", "Medium", "High"	Y
	HDMI-CEC	HDMI-CEC	RW	"Enable", "Disable"	Y
	Remote Power On / Off	RemotePowerOnOff	RW	"Enable", "Disable"	Y
	Function Keys mode	FunctionKeysMode	RW	"StandardF1F12mode", "SpecialFunctionMode"	Y
Advanced (Thunderbolt Configuration)	Devices under Thunderbolt	DevicesUnderThunderbolt	RW	"Enable", "Disable"	Y
	Security Level	SecurityLevel	RW	"NoSecurity", "UserAuthorization", "SecureConnect", "DisplayPortOnly"	Y

In TPM Control Utility (*4)		ShowTPMConfirmationMessage	RW	"Enable", "Disable"	N
		ShowTPMOwnerClearConfirmationMessage	RW	"Enable", "Disable"	N
In Toshiba Supervisor Registration Utility (*5)		ShowHDDPasswordMenu	RW	"Enable", "Disable"	N
In Toshiba Supervisor Registration Utility (*5)		OwnerString	RW	-	N

(*1) This item cannot be changed if TrustedExecutionTechnology is enabled.

(*2) To enable this item, it is required to set TPM to Enabled and VertualizationTechnology to VT-xAndVT-d.

(*3) This item takes several values as arguments.

(*4) This setting cannot be access through the BIOS Setup Utility. It can only be accessed from the TPM Control Utility or through the Toshiba WMI Interface.

(*5) Toshiba Supervisor Registration Utility, which is a DOS based utility. There is no access to these settings through the BIOS Setup Utility. These settings can only be accessed through the Toshiba Supervisor Registration Utility or the Toshiba WMI Interface.

(*6) Due to MS Logo policy, disabling “Secure Boot” is not allowed over WMI. The only way to disable “Secure Boot” is to open the BIOS Setup Utility screen and change this setting manually.

5 Special features for the supervisor

There are special features included in the BIOS that are only active when a supervisor password is set. This includes access to enable/disable certain built-in components and to restrict specific boot devices.

The Toshiba Supervisor Registration Utility, is a standalone utility for modifying the settings intended for the supervisor. The names of the settings in the first column are the names how they are used in the Supervisor Registration Utility. The second column contains the names of the settings accessed through the WMI interface.

Table 12 Settings for the supervisor

Name in Toshiba Supervisor Registration Utility	WMI Item name	Acceptable values	Description of the setting
RegistPswd	RegisterPassword	"Enable", "Disable"	Allow user to register user password
DeletePswd	DeletePassword	"Enable", "Disable"	Allow user to delete user password
ChangePswd	ChangePassword	"Enable", "Disable"	Allow user to change user password
NoLockPswd	NeverLockPassword	"Enable", "Disable"	Does not lock user password even if user password verification exceeds max retry counts
NoReqRgPswd	NeverForceRegisterPassword	"Enable", "Disable"	Special customer modified BIOS required If set to "Disable" BIOS will request the user to register or change user password on next boot
MaxChekTry	MaxTryCount	"1"-"15", "Unlimited"	Max retry count to verify user PW (1-15 or Unlimited). The default value is “3”.
MinPswdLen	MinimumPasswordLength	"1"-"15"	Minimum length of user password (1-15). The default value is “1”.
BiosSetup	BIOSSetup	"Enable", "Disable"	Allow to edit BIOS SETUP (SYSTEM SETUP)
BiosUpdate	BIOSUpdate	"Enable", "Disable"	Allow user to update the BIOS
NotViewMode	ViewMode	"Enable", "Disable"	“Enable”: User can only read but cannot write BIOS settings
RegDelHDDpw	RegisterHDDPassword	"Enable", "Disable"	Allow user to register a HDD password (user or master)
ChangeHDDpw	ChangeHDDPassword	"Enable", "Disable"	Allow user to change the HDD password
S4LockHDDpw	HDDAutoUnlockS4	"Enable", "Disable"	Special customer modified BIOS required
S5LockHDDpw	HDDAutoUnlockS5	"Enable", "Disable"	Special customer modified BIOS required
ActivateTPM	ActivateTPM	"Enable", "Disable"	Allow to configure “TPM Enable/Disable” in BIOS SETUP
OwnerClrTPM	OwnerClearTPM	"Enable", "Disable"	Allow to configure “Clear TPM Owner” in BIOS SETUP
BTcert	BluetoothAuthentication	"Enable", "Disable"	Enable/disable BlueTooth authentication
FPcert	FingerprintAuthentication	"Enable", "Disable"	Enable/disable FingerPrint authentication
CreateToken	CreateToken	"Enable", "Disable"	Special customer modified BIOS required Allow to create user token
RemoveToken	RemoveToken	"Enable", "Disable"	Special customer modified BIOS required

Name in Toshiba Supervisor Registration Utility	WMI Item name	Acceptable values	Description of the setting
			Allow to delete user token
IoCOM	DeviceSerialPort	"Enable", "Disable"	Enable/disable Serial Port (RS-232C serial port)
IoODD	DeviceODD	"Enable", "Disable"	Enable/disable Optical Disc Drive (internal CD-ROM drive, CD/DVD/HD-DVD/BD multi-drive)
Io2ndHDD	Device2ndHDD	"Enable", "Disable"	Enable/disable Second Hard Disk Drive
IoBluetooth	DeviceBluetooth	"Enable", "Disable"	Enable/disable Bluetooth (except for SD/USB Bluetooth)
IoMODEM	DeviceModem	"Enable", "Disable"	Enable/disable Internal Modem
IoUSB	DeviceUSB	"Enable", "Disable"	Enable/disable USB Connector
IoPCCard	DevicePCCard	"Enable", "Disable"	Enable/disable PC Card Slot (disabling this item, boot from a PC Card ATA is also disabled)
IoSD	DeviceSDCard	"Enable", "Disable"	Enable/disable SD Card Slot (disabling this item, boot from SD memory card is also disabled)
IoIEEE1394	DeviceIEEE1394	"Enable", "Disable"	Enable/disable i.LINK(IEEE1394) Connector
IoExpCard	DeviceExpresscard	"Enable", "Disable"	Enable/disable xpressCard Slot
IoWiredLAN	DeviceWiredLAN	"Enable", "Disable"	Enable/disable Internal Wired LAN
IoWlessLAN	DeviceWirelessLAN	"Enable", "Disable"	Enable/disable Internal Wireless LAN
IoWlessWAN	DeviceWirelessWAN	"Enable", "Disable"	Enable/disable Internal Wireless WAN
IoMediaSlot	DeviceMediaSlot	"Enable", "Disable"	Enable/disable Internal Media Slot
IoESATA	DeviceESATA	"Enable", "Disable"	Enable/disable eSATA connector (or eSATA portion of an eSATA+USB connector)
IoWebcam	DeviceWebcam	"Enable", "Disable"	Enable/disable Internal Webcam
EnSmartCard	DeviceSmartCard	"Enable", "Disable"	Enable/disable SmartCard device
Boot1stHDD	Boot1stHDD	"Enable", "Disable"	Enable/disable Boot from 1st Hard Disk Drive
Boot2ndHDD	Boot2ndHDD	"Enable", "Disable"	Enable/disable Boot from 2nd Hard Disk Drive
BootODD	BootODD	"Enable", "Disable"	Enable/disable Boot from Optical Disc Drive
BootFDD	BootFDD	"Enable", "Disable"	Enable/disable Boot from Floppy Disk Drive
BootLAN	BootLAN	"Enable", "Disable"	Enable/disable Boot from LAN
BootUSB	BootUSB	"Enable", "Disable"	Enable/disable Boot from USB Memory (USB flash drive and USB Hard Disk Drive)
BootESATA	BootESATA	"Enable", "Disable"	Enable/disable Boot from eSATA device

Table 13 Note on some of the Settings

Name in Toshiba Supervisor Registration Utility	WMI Item name	Comment
NotViewMode	ViewMode	NotViewMode(Enabled) = ViewMode(Disabled) NotViewMode(Disabled) = ViewMode(Enabled)
S4LockHDDpw	HDDAutoUnlockS4	S4LockHDDpw(Enabled) = HDDAutoUnlockS4(Disabled)
S5LockHDDpw	HDDAutoUnlockS5	S5LockHDDpw(Enabled) = HDDAutoUnlockS5(Disabled)

Note: The default value of the setting is underlined.

Note: If NeverLockPassword is disabled and the user enters the user password incorrectly too many times the computer locks. The computer can only then be unlocked by entering the Supervisor Password.

6 Troubleshooting

Should you encounter any remote access problems, please try the following solutions.

Checking DCOM permissions

1. Open **Component Service** by opening **Start** -> **Run** and type in **Dcomcnfg**.
2. Expand **Component Service** -> **Computers** -> **My computer**.
3. Go to the properties of **My Computer**.
4. Select the **COM Security** tab.
5. Click **Edit Limits** under **Access Permissions**, and ensure **Everyone** user group has **Local Access** and **Remote Access** permission.
6. Click **Edit Limit** for the launch and activation permissions, and ensure **Everyone** user group has **Local Activation** and **Local Launch** permission.
7. Highlight **DCOM Config** node, and right click **Windows Management and Instruments**, then click **Properties** and check that the used credentials have remote access rights for all options.

Checking permissions for the used credentials to the WMI namespace

1. Click **Start** -> **Run** and type in **WMImgmt.msc**, and then click **OK**.
2. Right click **WMI Control**, then click **Properties**.
3. Go to the **Security** tab.
4. Select **Root** and click **Security**.
5. Ensure **Authenticated Users** has **Execute Methods**, **Provider Right** and **Enable Account** permission; ensure **Administrators** has all permissions.

Verify WMI Impersonation Rights

1. Click **Start** -> **Run** and type in **gpedit.msc**, and then click **OK**.
2. Under **Local Computer Policy**, expand **Computer Configuration** -> **Windows Settings**.
3. Expand **Security Settings** -> **Local Policies**, and click **User Rights Assignment**.
4. Verify that the SERVICE account is specifically granted for **Impersonate a client after authentication**.

Check Network access sharing and security model

1. Click **Start** -> **Run** and type in **secpol.msc**, and then click **OK**.
2. Expand **Local Policies** -> **Security Options**.
3. Check if **Network Access: Sharing security model for local accounts** is set to “Classic”.

Check Firewall settings

Please refer to the following webpage for more details:
http://msdn.microsoft.com/en-us/library/windows/desktop/aa389286%28v=vs.85%29.aspx

Some general information about remote access

Please refer to the following webpage for more details:
http://msdn.microsoft.com/en-us/library/windows/desktop/aa389290%28v=vs.85%29.aspx

Some general infos about access issues in combination with UAC

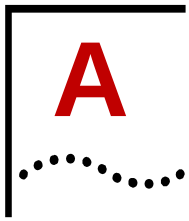
Please refer to the following webpage for more details:
http://msdn.microsoft.com/en-us/library/windows/desktop/aa826699%28v=vs.85%29.aspx

Special note for Windows 8

With Windows 8, the UAC settings have been changed. WMI remote access for local Admins is restricted by default and only Domain Admins have guaranteed access.

For testing purpose or in a Workstation environment, follow the steps below to disable UAC for remote Admins:

1. Click **Start** -> **Run** and type in **regedit**, and then click **OK**.
2. Expand registry folder:
   HKEY_LOCAL_MACHINE¥SOFTWARE¥Microsoft¥Windows¥CurrentVersion¥Policies¥System
3. Creade a 32-bit **DWord** entry named **LocalAccountTokenFilterPolicy** if not existing.
4. Set **Value** to **1** to disable UAC for remote Adminsistrators.



Visual Basic script to set a password on a remote computer

Please save this script into a file called procedures.vbs. The following script will load this script into memory and use the function ***SetConfigurationMode.***

```
'
'    Function   : authenticate/deauthenticate with
Supervisor privilege
'   parameter 1 : WMI service object
'             2 : input parameter value for mode control
method
'
Function SetConfigurationMode(objWMIService,
strInParamValue)
    Dim colItems, objItem

    'executes a WQL query
    Set colItems = objWMIService.ExecQuery("Select * from
ModeControl where InstanceName='ACPI\\PNP0C14\\0_0'")

    For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetConfigurationMode strInParamValue, strReturn
    Next

    SetConfigurationMode = strReturn

End Function
```

This script can be used for setting, changing or deleting passwords on a remote computer.

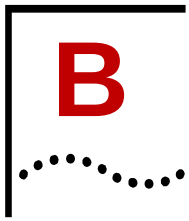
```
'
'Sample VBScript: Set/Change/Delete a password on a remote
computer.(restrictions: cannot set supervisor password, but
can modify or delete)
'
'   command line: cscript.exe SetPasswordRemote.vbs [type]
"[scrambled old password]" "[scrambled new password]"
["[scrambled old password]" "[scrambled new password]"]
"[scrambled SupervisorPassword]" [ComputerName|IPAddress]
[UserName] [Password]
'   argument  1 : password type - user:        use if you
want to change BIOS User Password,
'                                 supervisor:  use if you
want to change BIOS Supervisor Password,
'                                 userHDD:     use if you
want to change User Only HDD Password,
'                                 master+user: use if you
want to change Master + User HDD Password
'                 Master + User HDD Password requires 4
passwords, the first 2 for Master HDD Password and the
another 2 for User HDD Password
'             2 : the scrambled 1st old password with
quotes (Master HDD Password if choosing Master + User HDD
Password)
'             3 : the scrambled 1st new Password with
quotes (Master HDD Password if choosing Master + User HDD
Password)
```

```
'             4(optional) : the scrambled 2nd old password
with quotes (User HDD Password if choosing Master + User
HDD Password)
'             5(optional) : the scrambled 2nd new password
with quotes (User HDD Password if choosing Master + User
HDD Password)
'             6 : scrambled supervisor password with quotes
'             7 : the IP, the FQDN, or the Computer name of
the client PC you want to access
'             8 : a username that has Administrator
privileges on the client PC you want to access
'             9 : password for the username that has
Administrator privileges on the client PC you want to
access

'declare application name
Dim strAppName
strAppName = "SetPasswordRemote"

On Error Resume Next
Dim argcount, strType, objWMIService, objItem, colItems,
strComputer, strInParamValue, strReturn, strStatus,
strFileName
Dim strOldPassword1, strNewPassword1, strOldPassword2,
strNewPassword2, strSupervisorPassword, strParameter,
strUserName, strPassword

'create Object to open the procedure file
strFileName = "procedures.vbs"
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.OpenTextFile(strFileName, 1)   '1 -
for reading
If objFile is Nothing Then
    WScript.Echo "You can not run this application without
file " & strFileName
    WScript.Quit
End If
Execute objFile.ReadAll()


''check input parameters
argcount = WScript.Arguments.Count
If argcount <> 7 And argcount <> 9 Then
    ShowUsage()
    WScript.Quit
End If

'define variables
If argcount = 7 Then
    Select Case WScript.Arguments(0)
    Case "user"
        strType = "UserPassword"
    Case "supervisor"
        strType = "SupervisorPassword"
    Case "userHDD"
        strType = "UserOnlyHDDPassword"
    Case Else
        ShowUsage()
        WScript.Quit
    End Select

    strOldPassword1 = WScript.Arguments(1)
```

```
    strNewPassword1 = WScript.Arguments(2)
    strInParamValue = strType + "," + strOldPassword1 + ","
+ strNewPassword1 + ";"
    strSupervisorPassword = WScript.Arguments(3)
    strComputer = WScript.Arguments(4)  'computer name or
computer's IP address
    strUserName = WScript.Arguments(5)
    strPassword = WScript.Arguments(6)
Else  'argcount is 9
    Select Case WScript.Arguments(0)
    Case "master+user"
        strType = "Master+UserHDDPassword"
    Case Else
        ShowUsage()
        WScript.Quit
    End Select

    strOldPassword1 = WScript.Arguments(1)
    strNewPassword1 = WScript.Arguments(2)
    strOldPassword2 = WScript.Arguments(3)
    strNewPassword2 = WScript.Arguments(4)
    strInParamValue = strType + "," + strOldPassword1 + ","
+ strNewPassword1 + "," + strOldPassword2 + "," +
strNewPassword2 + ";"
    strSupervisorPassword = WScript.Arguments(5)
    strComputer = WScript.Arguments(6)  'computer name or
computer's IP address
    strUserName = WScript.Arguments(7)
    strPassword = WScript.Arguments(8)

End If



wbemImpersonationLevelImpersonate = 3
wbemAuthenticationLevelPktPrivacy = 6

'get the locator object
Set objSWbemLocator =
CreateObject("WbemScripting.SWbemLocator")

'get the service object from the remote server
Set objWMIService =
objSWbemLocator.ConnectServer(strComputer, "root\WMI",
strUserName, strPassword)

If Err.Number <> 0 Then
    WScript.Echo "Unable to connect to " & strComputer &
": " & Hex(Err.Number) & "."
    WScript.Quit
End If

objWMIService.Security_.ImpersonationLevel =
wbemImpersonationLevelImpersonate
objWMIService.Security_.AuthenticationLevel =
wbemAuthenticationLevelPktPrivacy


If strSupervisorPassword <> "" Then
    ''authenticate with Supervisor privilege
    strParameter = "Start," + strSupervisorPassword + ";"
```

```
    strReturn = SetConfigurationMode(objWMIService,
strParameter)
    If strReturn <> 0 Then
        WScript.Echo "Supervisor password authentication
failed. Error: " & GetErrMsg(Hex(strReturn))
        WScript.Quit
    Else
        WScript.Echo "Supervisor password successfully
authenticated."
    End If
End If

'executes a WQL query
Set colItems = objWMIService.ExecQuery("Select * from
Password where InstanceName='ACPI\\PNP0C14\\0_0'")

''modify the supervisor password
For Each objItem in colItems
    'execute the method and obtain the return status
    objItem.SetPassword strInParamValue, strReturn
Next

WScript.Echo strAppName & ": " & GetErrMsg(Hex(strReturn))


''deauthenticate from supervisor mode
If strType = "SupervisorPassword" And strReturn = 0 Then
    strParameter = "End," + strNewPassword1 + ";"
'process by using new supervisor password if success of
changing password
Else
    strParameter = "End," + strSupervisorPassword + ";"
End If

If strParameter <> "End,;" Then     'if there is no
supervisor password, skip deauthentication process
    strReturn = SetConfigurationMode(objWMIService,
strParameter)
    If strReturn <> 0 Then
        WScript.Echo "Supervisor password deauthentication
failed. Error: " & GetErrMsg(Hex(strReturn))
        WScript.Quit
    Else
        WScript.Echo "Supervisor password successfully
deauthenticated."
    End If
End If

WScript.Quit



''usage help
Sub ShowUsage()
    WScript.Echo "Usage: cscript.exe SetPassword.vbs [type]
""[scrambled old password]"" ""[scrambled new password]""
[""[scrambled old password]"" ""[scrambled new password]""]
""[scrambled SupervisorPassword]"" [ComputerName|IPAddress]
[UserName] [Password]"
    WScript.Echo "       [type]      user
use if you want to change BIOS User Password"
```

```
    WScript.Echo "                  supervisor
use if you want to change BIOS Supervisor Password"
    WScript.Echo "                  userHDD
use if you want to change User Only HDD Password"
    WScript.Echo "                  master+user
use if you want to change Master + User HDD Password"
    WScript.Echo "        Master + User HDD Password
requires 4 passwords, the first 2 for Master HDD Password
and the another 2 for User HDD Password"
End Sub


''convert an error code to a string
Function GetErrMsg(err)
    Dim strMsg
    Select Case err
        Case "0"
           strMsg = "The operation was successful."
        Case "8004100C"
           strMsg = "Feature or operation is not
supported."
        Case "80041008"
           strMsg = "One of the parameters to the call is
not correct."
        Case "80041003"
           strMsg = "Write Protect error"
        Case "80041062"
           strMsg = "Operation failed because the client
did not have the necessary security privilege."
        Case "80045001"
           strMsg = "Authentication failure."
        Case "80045002"
           strMsg = "Password not registered."
        Case Else
           strMsg = "error code " + err
    End Select

    GetErrMsg = strMsg
End Function
```



Sample scripts in PowerShell

Some general notes on PowerShell

By default, Windows rejects to execute downloaded unsigned power shell scripts.

To run below example scripts please do the following:

Open a **Command** window with Administrator privilege and type in *powershell set–executionpolicy remotesigned.*

In the CMD window, start Powershell scripts with *powershell .\scriptname.ps1*

Source: http://technet.microsoft.com/en-us/library/ee176961.aspx

Read all BIOS settings and output to the console

```
#
# Below sample script will display all available BIOS settings on the standard output
# Format: Item Name, Accessibility, Current Value
#

# Replace “Computername” with the IP, the FQDN, or the Computer name of the client PC you want to access.
# Use “localhost” or remove this line if you want to access the local computer
$strComputername = “Computername”

# Replace “Username” with a username that has Administrator privileges on the client PC you want to access.
# Use the domain administrator username to access a client PC belonging to an Active Directory domain.
# Leave "Username" blank to get prompted for a username
# To avoid getting prompted for the password, remove the authentication part from the script and execute the script under Administrator or Domain Administrator privileges.
$cred = get-credential “Username”

gwmi –namespace "root\wmi" –class "QueryBiosSettings" –credential $cred –computer $strComputername | ForEach-Object {if($_.CurrentSetting -ne "") {Write-Host $_.CurrentSetting}}
```

Write a single BIOS item

```
#
# Below script will change a single BIOS settings
#

# Replace “Computername” with the IP, the FQDN, or the Computer name of the client PC you want to access.
# Use “localhost” or remove this line if you want to access the local computer
$strComputername = "Computername"
```

```
# Replace “Username” with a username that has Administrator
privileges on the client PC you want to access.
# Use the domain administrator username to access a client
PC belonging to an Active Directory domain.
# Leave "Username" blank to get prompted for a username
# To avoid getting prompted for the password, remove the
authentication part from the script and execute the script
under Administrator or Domain Administrator privileges.
$cred = get-credential “Username”

# Please input Passwords as scrambled keyboard pass code
# Use the Generator utility to convert plain Passwords into
scrambled keyboard pass codes
$SvPW = ‘Password’

# Input the Item name. E.g. SleepAndCharge
# Consult the manual or use the "QueryBiosItems" Class to
get a list of all available BIOS settings
$Item = ""

# Input the parameter. E.g. AutoMode
# Consult the manual or use the
"QueryBiosSettingsParameterr" Class to get a list of all
possible parameter
# Please note that all values are case sensitive
$value = ""

Write-Host "-***-`n"

# Authenticate
$result=""
$mode= gwmi  –namespace "root\wmi" -class "ModeControl" –
credential $cred –computer $strComputername | where
{$_.InstanceName -match "ACPI\\pnp0c14\\0.0"}
$result = $mode.SetConfigurationMode("Start,$SvPW;").Return

if ($result -eq 0)
    {
    Write-Host "Successfully authenticated `n"



    $result=""
    $read = gwmi  –namespace "root\wmi" -class
"BiosSetting" –credential $cred –computer $strComputername
| where {$_.InstanceName -match “ACPI\\pnp0c14\\0.0"}

    # Filter the line for the parameter
    $currentSetting =
$read.GetBiosSetting($Item+";").CurrentSetting.Split(",",3)
[2]+";"
    Write-Host "Current Setting =" $currentSetting

    #Set new value
    $result = $read.SetBiosSetting("$Item,$value;").Return
    if ($result -eq 0)
        {
        Write-Host "Successfully applied the new setting"
        $currentSetting =
$read.GetBiosSetting($Item+";").CurrentSetting.Split(",",3)
[2]+";"
        Write-Host "New Setting = $currentSetting `n"
```

```
        }

    else {Write-Host "Fail, Error Code= $result"}

    # Deauthenticate
    $result=""
    $result = $mode.SetConfigurationMode("End"
+",$SvPW;").Return
    if ($result -eq 0) {Write-Host "Sucessfully
deauthenticated"}
    else {Write-Host "Deauthentication failed, Error Code=
"$result}

    Write-Host "`nPlease perform a reboot to apply the
setting"
    }

else {Write-Host "Authentication  failed, Error Code=
"$result}


Write-Host "`n-***-"
```

Save current BIOS settings to a file

```
#
# Below script is saving the current Bios settings into a
file
# After changing settings, the script
"ReadSavedBiosSettingsfromFileandWriteToBios.ps1" can be
used to write the modified settings back into the Bios
#

# Replace “Computername” with the IP, the FQDN, or the
Computer name of the client PC you want to access.
# Use “localhost” or remove this line if you want to access
the local computer
$strComputername = “Computername”

# Replace “Username” with a username that has Administrator
privileges on the client PC you want to access.
# Use the domain administrator username to access a client
PC belonging to an Active Directory domain.
# Leave "Username" blank to get prompted for a username
# To avoid getting prompted for the password, remove the
authentication part from the script and execute the script
under Administrator or Domain Administrator privileges.
$cred = get-credential “Username”


# Read the raw data
$list = ""
$rawlist = gwmi –namespace "root\wmi" –class
"QueryBiosSettings" –credential $cred –computer
$strComputername

foreach ($item in $rawlist)
      {
      # Remove empty rows and not supported functions
      if(($item.CurrentSetting -ne "") -and
($item.CurrentSetting -notlike "*NotSupported"))
```

```
                {
                $list += $item.CurrentSetting +"`n"
                }
        }


# Replace "CurrentBiosSettings.txt" with your path and
filename
Set-Content CurentBiosSettings.txt $list

Write-Host "Current Bios Settings successfully saved"
```

Reads saved BIOS settings from a file and writes it back to BIOS

```
#
# Below script is reading Bios settings from a file and
writing them back into the Bios
# Together with the previous script it can be used to
realize a backup function but also to modify a bunch of
Bios settings in one loop
#

# Each Bios item should be present in one line.  You need
to follow this structure:
# BiosItemA, settingA1, settingA2, ....
# BiosItemB, settingB1, settingB2, ....
# BiosItem.......
#
# Alternatively, the output of the script
"SaveCurrentBiosSettingsToFile.ps1" can be used as a base


# Replace “Computername” with the IP, the FQDN, or the
Computer name of the client PC you want to access.
# Use “localhost” or remove this line if you want to access
the local computer
$strComputername = "Computername"

# Replace “Username” with a username that has Administrator
privileges on the client PC you want to access.
# Use the domain administrator username to access a client
PC belonging to an Active Directory domain.
# Leave "Username" blank to get prompted for a username
# To avoid getting prompted for the password, remove the
authentication part from the script and execute the script
under Administrator or Domain Administrator privileges.
$cred = get-credential “Username”

# Replace "CurrentBiosSettings.txt" with your path and
filename
$file = "CurentBiosSettings.txt"

# Please input Passwords as scrambled keyboard pass code
# Use the Generator utility to convert plain Passwords into
scrambled keyboard pass codes
$SvPW = ‘Password’

# Remove the troublemaking empty line from the end of the
input file, generated but the Power Shell Set-Content
function.
$list = gc $file | where {$_ -ne ""}
```

```
Write-Host "-***-`n"

# Authenticate
# Notice: backslash must be escaped by "\"
$result = ""
$mode= gwmi  –namespace "root\wmi" -class "ModeControl" –
credential $cred –computer $strComputername | where
{$_.InstanceName -match "ACPI\\pnp0c14\\0.0"}
$result = $mode.SetConfigurationMode("Start,$SvPW;").Return

if ($result -eq 0)
    {
    Write-Host "Successful Authenticated"

    # Get access to the Bios Setting Class
    $function = gwmi  –namespace "root\wmi" -class
"BiosSetting" –credential $cred –computer $strComputername
| where {$_.InstanceName -match “ACPI\\pnp0c14\\0.0"}

    $result = ""
    foreach ($item in $list)
      {
             # Split the input line into 3 substrings
            # string[0] = Parameter
            # string[1] = Read/Write indicator
            # string[2] = The parameter value(s) separated
by comma
            $para =
$item.Split(",")[0]+","+$item.Split(",",3)[2]+";"

             # Check if setting is writable and restore
setting if writable
            if ($item.Split(",")[1] -ne "RO")
                  {
                  $result =
$function.SetBiosSetting("$para").Return
                   if ($result -eq 0)
                                  {
                                 Write-Host "Successfully
restored setting for: "$item.Split(",")[0]": Setting =
"$item.Split(",",3)[2]"`n"
                                  }
                           else
                                  {
                                  Write-Host "Could not
write the value, maybe there are dependencies with other
settings: "$item.Split(",")[0]". Error Code: "$result"`n"
                                  }
                   }
        }

       #Deauthentication
       $result = ""
       $result = $mode.SetConfigurationMode("End"
+",$SvPW;").Return
       if ($result -eq 0) {Write-Host "Successfully
deauthenticated"}
       else {Write-Host "Deauthentication failed, Error
Code= "$result
       }
```

```
    Write-Host "`nPlease perform a reboot to apply the
setting"
    }

else {Write-Host "Authentication  failed, Error Code=
"$result}

Write-Host "`n-***-"
```

Set or Change BIOS Passwords

```
#
# Below is a sample script to change or delete the
Supervisor or User Password
# Please notice that for security reasons, the Supervisor
Password must be set manual or by a separate tool initially
(e.g. WinPE based utility) to be able to access the Bios
Passwords remotely
#

# Replace “Computername” with the IP, the FQDN, or the
Computer name of the client PC you want to access.
# Use “localhost” or remove this line if you want to access
the local computer

$strComputername = “Computername”
# Replace “Username” with a username that has Administrator
privileges on the client PC you want to access.
# Use the domain administrator username to access a client
PC belonging to an Active Directory domain.
# Leave "Username" blank to get prompted for a username
# To avoid getting prompted for the password, remove the
authentication part from the script and execute the script
under Administrator or Domain Administrator privileges.
$cred = get-credential “Username”

# Define the access level (User or Supervisor)
# - SupervisorPassword
# - UserPassword
# Change below  to "UserPassword" for changing / setting /
deleting the User Password
$Item = "SupervisorPassword"

# Please input Passwords as scrambled keyboard pass code
# Use the Generator utility to convert plain Passwords into
scrambled keyboard pass codes
# Set new Password blank to delete the Password
$oldPW = ‘oldPassword’
$newPW = ‘newPassword’

# Please input below the actual Supervisor Password.
# When changing the Supervisor password, $oldPW and $SvPW
are identical.
$SvPW = ‘Password’

Write-Host "-***-`n"

# Authenticate with Supervisor privilege
$result = ""
```

```
$mode= gwmi  –namespace "root\wmi" -class "ModeControl" –
credential $cred –computer $strComputername | where
{$_.InstanceName -match "ACPI\\pnp0c14\\0.0"}
$result = $mode.SetConfigurationMode("Start,$SvPW;").Return

if ($result -eq 0)
    {
    Write-Host "Successful Authenticated"

    # Access function
    $function = gwmi  –namespace "root\wmi" -class
"Password" –credential $cred –computer $strComputername |
where {$_.InstanceName -match "ACPI\\pnp0c14\\0.0"}

    # Set or change the Password
    $result= ""
    $result =
$function.SetPassword("$Item,$oldPW,$newPW;").Return
    if ($result -eq 0) {Write-Host "Successfully changed
the Password"}
    else {Write-Host "Fail, could not change the Password,
Error Code= "$result}


    $result =""
    # adjust de-authentication PW if access level is
Supervisor
    if($item -like "SupervisorPassword") {$SvPW = $newPW}

    # Deauthenticate
    $result = $mode.SetConfigurationMode("End"
+",$SvPW;").Return
    if ($result -eq 0) {Write-Host "Sucessfully
deauthenticated"}
    else
        {
        # Check if $newPW is empty. If so, it means it is
removed
        if ($newPW -eq "")
            {
            Write-Host "Password removed"
            }
        Else
            {
             Write-Host "Deauthentication failed, Error
Code= "$result
            }
        }

    Write-Host "`nPlease perform a reboot to apply the
setting"
    }

else {Write-Host "Authentication failed, Error Code=
"$result}
```