Enforce BitLocker Encryption Worklet

This worklet describes how to keep your drives encrypted with ongoing evaluation and remediation (Windows 8 and Windows 10).

For detailed steps on creating and scheduling a worklet, refer to:

Creating a Worklet

For an overview on Worklets, see this article:

How to Use Worklets

The Worklets

The following evaluation and remediation worklets can be used to enforce BitLocker encryption.

Evaluation Code

```
# PowerShell 4.0 and Above
# Windows 8 and later

#Get BitLocker status for All Drives
try { $encryption = Get-BitLockerVolume -ErrorAction Stop }
catch { Write-Output "Unable to determine BitLocker status" }

# Count Drives and initialize lists for later output
$numDrives = $encryption.Count
$encCount = 0
$encrypted = @()
$unencrypted = @()

# Loop through each drive and see if it is Protected or Note
# Add to the appropriate list, Encrypted or Unencrypted
foreach ($drive in $encryption) {
    $encStatus = $drive.ProtectionStatus
    $encInProgress = $drive.VolumeStatus
    if ( ($encStatus -match 'On') -or ($encInProgress -match "EncryptionInProgr
ess") ) {
        $encrypted += $drive.MountPoint
        $encCount++
    } else {
        $unencrypted += $drive.MountPoint
    }
}

# Output drive statuses so the can be seen in the Activity Log
Write-Output "Encrypted Drives: $encrypted`n"
Write-Output "Unencrypted Drives: $unencrypted`n"

# Determine Compliant based on if the number of Encrypted
# Drives matches the number of Total Drives
if ($encCount -eq $numDrives) {
    Write-Output "Compliant"
    exit 0
} else {
    Write-Output "Non-Compliant"
    exit 1
}
```

This evaluation requests BitLocker status for all physical disk drives on the target device. It then compares the count of encrypted drives to the total number present. If all drives are encrypted then it returns Compliant (exit 0), otherwise, it returns Non-Compliant (exit 1).

Remediation Code

```
# PowerShell 4.0 and Above
# Windows 8 and later

# Define where you want your Recovery Key to be exported
# Note that this needs to be a local (non-network) drive.
$keyPath = 'C:\temp'

$toEncrypt = Get-BitLockerVolume | Where-Object { $_.VolumeStatus -match 'Decry
pted' }

# Loop through each Unencrypted Drives
# Enable Bitlocker and Export their Recovery Keys
foreach ( $drive in $toEncrypt ) {
    $driveLetter = $drive.MountPoint.Replace(':','')
    try {
        #Enable Bitlocker
        Enable-BitLocker -MountPoint $driveLetter -EncryptionMethod Aes128 -Rec
overyPasswordProtector | Out-Null

        #Export Key and Key ID to a File
        $recID = (Get-BitLockerVolume -MountPoint $driveLetter).KeyProtector.Ke
yProtectorID
        $recKey = (Get-BitLockerVolume -MountPoint $driveLetter).KeyProtector.R
ecoveryPassword
        Set-Content -Path "$keyPath\BitlockerRecoveryKey_$driveLetter.txt" -For
ce -Value "Recovery Key ID: $recID"
        Add-Content -Path "$keyPath\BitlockerRecoveryKey_$driveLetter.txt" -Val
ue "Recovery Key: $recKey"
    } catch {
        Write-Output "Unable to Encrypt $($drive.MountPoint)"
    }
}
```

The remediation code has one editable variable ($keyPath). Use this to define where the recovery key will be stored. The recovery key is necessary to decrypt the drive should that become necessary in the future.

This worklet initially runs a similar check as the evaluation code to enumerate each physical drive that is not encrypted. Using this information, it starts encryption on each of these drives and exports the recovery key to a text file in the previously specified location.