**File Integrity Monitor**

File Integrity Monitor (FIM) is a program or software that checks files to identify tampering and cyber-attacks. This monitoring task is performed periodically to ensure that the files have not been compromised. A FIM first creates a baseline of all the files to be monitored, this baseline is a key value pair that contains the file name and its hash value generated using hashing functions. It then checks the files periodically regenerates the hash value and compares it against the existing baseline. Any change in the file would cause the new hash value to change and this would indicate that the file has been tampered with, in which case the FIM sends an alert to the concerned authority.

In this program, we will attempt to implement a FIM of our own using Powershell and test it against various scenarios. To create our FIM will follow the following flow chart.

**Flow chart**

1. Ask the user for the path to the file or directory to be scanned.

2. Check whether the specified object is a file or a directory

2.b. If it’s a directory recursively read all files in it and move on to the next step

2.a. IF it’s a file move on to the next step

3. Ask the user whether they want to create a new baseline or use an existing baseline.

3.b. If chosen to create a new base lime move on and create the new base line and update any existing baseline file. Go to step 4

3.a. If chosen to use the existing base line, check whether the file or files have an existing base line if yes go to step 5 else go to step 4

**No**

**Yes**

4. Create a new base line

5. Check the hash values of the files and compare them against the baseline values.

5.a. If the values are the same display “No Change Detected”

5.b. If the Values are not the same then display “Change detected” and send alert to the authorities

**PowerShell Program**

For testing purposes create a folder with 3 text files containing different values and save the path to the file.

* E:\Cyber security Projects\File Integrity Monitor\Test

Open powershell ISE and start a new program.

**File Integrity Monitor Script**

#Add the presentation framework for a windows message box

Add-Type -AssemblyName PresentationFramework

#Function to populate\_dictionary

Function populate\_dictionary()

{

#retrieve the value from baseline.txt

$dict1 = @{}

$cur\_baseline = Get-Content "E:\Cyber security Projects\File Integrity Monitor\baseline.txt"

foreach ($l in $cur\_baseline)

{

$dict1.add($l.Split("|")[0],$l.Split("|")[1])

}

return $dict1

}

#Function to create a new base line

Function create\_new\_baseline($file\_arr)

{

foreach ($f in $file\_arr)

{

$hash = calculate\_hash\_value $f.FullName

#display the hash object. it contains 3 values Algorithm, Path and Hash.

#Retrieve the Path and Hash vales and store them in the baseline.txt file. Here Path is the key and Hash is the value

$k = $hash.Path

$v = $hash.Hash

"$k|$v"| Out-File -FilePath "E:\Cyber security Projects\File Integrity Monitor\baseline.txt" -Append

}

}

#Function to calculate the hash value

Function calculate\_hash\_value($file)

{

$hash\_val = Get-FileHash -Path $file -Algorithm SHA512

return $hash\_val

}

#Function to check whether the baseline already exists

Function delete\_existing\_baseline()

{

$exists = Test-Path -Path "E:\Cyber security Projects\File Integrity Monitor\baseline.txt"

if ($exists -eq $true)

{

Remove-Item -Path "E:\Cyber security Projects\File Integrity Monitor\baseline.txt" -Force

}

else

{

echo "baseline does not exist. Creating a new Baseline"

}

}

Write-Host "Welcome to File Integrity Monitor" -ForegroundColor Green

Write-Host ""

#ask the user to provide the path to the file or folder that is to be checked

echo "Please enter the path to the file or folder that is to be checked."

Write-Host ""

$file\_path = Read-Host "Please specify the file path."

Write-Host ""

#Once the path has been received check whether it points to a file (leaf) or a directory (container)

Write-Host ""

$value = Test-Path -Path $file\_path -PathType Leaf

Write-Host ""

# here value is a variable that contains a boolean value. This would be true of the path points to a file and false otherwise

Write-Host ""

if ($value -eq $False)

{

# go through the directory to get the child items

$global:files = Get-ChildItem -Path $file\_path

Write-Host ""

echo "files found"

Write-Host ""

}

#once files have been identified ask the user whether he or she wants to create a new baseline or proceed with an existing baseline.

Write-Host "Please enter A to create a new Baseline"

Write-Host ""

Write-Host "Please enter B to proceed with existing baseline "

Write-Host ""

$usr\_input = Read-Host "Please enter your choice"

Write-Host ""

if ($usr\_input -eq "A".ToUpper())

{

delete\_existing\_baseline

create\_new\_baseline $files

#retrive contents of the new base line

$bl = Get-Content "E:\Cyber security Projects\File Integrity Monitor\baseline.txt"

#displaying the new baseline

echo $bl

}

elseif ($usr\_input -eq "B".ToUpper())

{

$new\_dict = populate\_dictionary

#check if a new file has been added

$child = Get-ChildItem -Path $file\_path

foreach ($c in $child)

{

$hash = calculate\_hash\_value $c.FullName

if ($new\_dict[$hash.Path] -eq $null)

{

echo "new file has been found"

delete\_existing\_baseline

create\_new\_baseline $child

}

else

{

$dict2 = populate\_dictionary

$x = calculate\_hash\_value $c.FullName

#if the hashes are different the value has been changed

#compare the existing hash in baseline.txt to a newly generated hash value

if ($x.Hash -eq $dict2[$x.Path])

{

echo $x.Path " hash value has not changed"

Write-Host ""

}

else

{

#displays a pop up message if there is achange in the hash value

[System.Windows.MessageBox]::Show($x.Path, ' hash value changed. File has been changed')

#echo $x.Path "hash value changed. File has been changed"

Write-Host ""

}

}

}

}