File Integrity Manager

Abstract

File Integrity Monitoring is one of the most basic security practices in which it's possible to check the integrity of files from the operating system and applications to detect, among other things, unauthorized modifications. It works by creating a baseline for all files to be monitored and then making constant comparisons of the actual state of a file with its baseline. Any anomalies, at file attribute and content, including permission level, will then trigger alerts for administrators on potential tampering and breach of security.

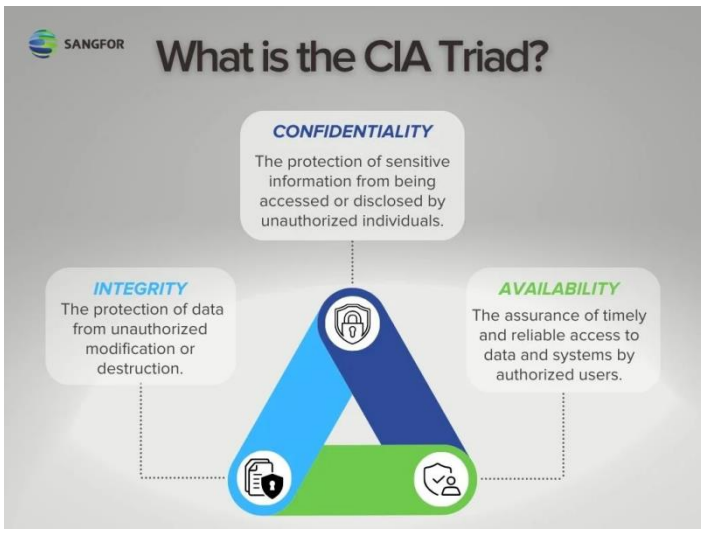
FIM is a very tremendous application for keeping confidentiality, integrity, and the availability of sensitive information. It aids one to respond swiftly to an incident and reduces the impact that a cyber attack could have through the detection of unauthorized changes. Furthermore, FIM is a very important tool for obtaining forensic insights by logging file access activities for post-incident analysis. It also provides support for file permissions and user privilege auditing in order to complete the access control functionality.

It proposes an efficient FIM solution, combining both offline with online monitoring approaches and dynamic inspection scheduling based on file classification. The system will grow in effectiveness and efficiency as the rationale created ensures that only high-risk files are monitored frequently. Preliminary testing results suggest that the approach described can significantly enhance current FIM methods.

Introduction

File integrity monitoring (FIM) is a critical cybersecurity mechanism that verifies the integrity of operating system and application files to detect unauthorized modifications, which could indicate malware infection, insider threats, or external attacks. This project presents a **Powershell-based** FIM solution that utilizes hash functions to ensure file integrity in designated directories. The core functionality is encapsulated in two functions: **Calculate-File-Hash**, which computes the **SHA384** hash of files, and **Erase-hashed-If-Already-Exists**, which manages existing hash records to prevent data overlap. Users can either collect new hashes or monitor existing files against previously stored hashes. The monitoring process operates continuously, checking for new files, modifications, or deletions, and providing real-time notifications for any detected changes. By combining offline and online monitoring approaches with dynamic inspection scheduling based on file classification, the proposed solution aims to enhance the efficiency and effectiveness of file integrity monitoring. Preliminary testing indicates significant improvements in performance, accuracy, ease of deployment, and management compared to existing FIM methods. This introduction outlines the fundamental components of the FIM solution, setting the stage for a detailed exploration of its design, implementation, and effectiveness in maintaining file integrity. The report is organized as follows: Section 2 provides an overview of FIM concepts, techniques, and use cases; Section 3 reviews related work and existing FIM solutions; Section 4 presents the design and architecture of the proposed system; Section 5 discusses the implementation and testing methodology; Section 6 analyzes the results and performance; and Section 7 concludes with key findings and future research directions.

CIA Triad



The **CIA triad** is an elementary structure of information security, based on three prime principles: **Confidentiality, Integrity, and Availability**. Each of these elements takes a very distinct role in protecting data and systems from unwanted access:

1. **Confidentiality:**

Confidentiality refers to a measure that allows ensuring sensitive information is not accessed by unauthorized people. It makes sure that the information shall only be accessed by the right people. Techniques for protecting confidentiality are **encryption**, **access control lists**, and **multi-factor authentication**. In a company, for example, access to financial records should be granted only to authorized personnel to avoid data leakage.

2. **Integrity:**

Integrity means that information must be accurate, consistent, and trustworthy throughout the life cycle of the data. This is the basis of data reliability, as unauthorized alterations may result in misinformation and even hinder decision-making practices in organizations. Herein, in a nutshell, this is integrity as far as **FIM** is concerned, where monitoring change mechanisms result in unauthorized change detection in files. For instance, a good **FIM** solution will raise an alarm to the administrators in case a file is modified maliciously or unintentionally for immediate remediation. Integrity techniques include **hashing algorithms**, **checksums**, and **audit logs**.

3. **Availability**:

Availability provides the correct access at the right time to the appropriate entity for the view of information. It is very essential for operational efficiency since system downtime disrupts business processes. Another strategies enhancing availability are **redundancy**, **failover systems**, and **regular maintenance to hardware and software**.

Implementation

```
Function Calculate-File-Hash($filepath) {
   $filehash = Get-FileHash -Path $filepath -Algorithm SHA512
   return $filehash
}
Function Erase-hashed-If-Already-Exists() {
   $hashedExists = Test-Path -Path .\hashed.txt

   if ($hashedExists) {
      # Delete the files
      Remove-Item -Path .\hashed.txt
   }
}
Write-Host "What would you like to do?"
Write-Host ""
Write-Host "    A) Collect new hashed?"
Write-Host "    B) Begin monitoring files with saved hashed?"
Write-Host ""
$response = Read-Host -Prompt "Please enter 'A' or 'B'"
Write-Host ""

if ($response -eq "A".ToUpper()) {
   # Delete hashed.txt if it already exists
   Erase-hashed-If-Already-Exists

   # Calculate Hash from the target files and store in hashed.txt
   # Collect all files in the target folder
   $files = Get-ChildItem -Path .\targetFiles
   # For each file, calculate the hash, and write to hashed.txt
   foreach ($f in $files) {
      $hash = Calculate-File-Hash $f.FullName
      "$($hash.Path)|$($hash.Hash)" | Out-File -FilePath .\hashed.txt -Append
   }

}

elseif ($response -eq "B".ToUpper()) {
   # Dictionary created to store the hashes
   $fileHashDictionary = @{}
# Load file|hash from hashed.txt and store them in a dictionary
   $filePathsAndHashes = Get-Content -Path .\hashed.txt
   foreach ($f in $filePathsAndHashes) {
       $fileHashDictionary.add($f.Split("|")[0],$f.Split("|")[1])
   }
   # Begin (continuously) monitoring files with saved hashed
   while ($true) {
      Start-Sleep -Seconds 1
      $files = Get-ChildItem -Path .\targetFiles
      # For each file, calculate the hash, and write to hashed.txt
      foreach ($f in $files) {
         $hash = Calculate-File-Hash $f.FullName
```

```
#"$($hash.Path)|$($hash.Hash)"|Out-File-FilePath .\hashed.txt -Append
        # Notify if a new file has been created
        if ($fileHashDictionary[$hash.Path] -eq $null) {
            # A new file has been created!
            Write-Host "$($hash.Path) has been created!" -ForegroundColor Green
        }
        else {

            # Notify if a new file has been changed
            if ($fileHashDictionary[$hash.Path] -eq $hash.Hash) {
                # The file has not changed
            }
            else {
                # File file has been compromised!, notify the user
                Write-Host "$($hash.Path) has changed!!!" -ForegroundColor Yellow
            }
        }
    }
    foreach ($key in $fileHashDictionary.Keys) {
        $hashedFileStillExists = Test-Path -Path $key
        if (-Not $hashedFileStillExists) {
            # One of the hashed files must have been deleted, notify the user
            Write-Host "$($key) has been deleted!" -ForegroundColor DarkRed -BackgroundColor Gray
        }
    }
  }
}
```

Result:

I created 4 test files (test1.txt, test2.txt, test3.txt, test4.txt) for testing FIM. During the test, I changed the contents of test2.txt file and copied test4 (test4-Copy.txt).

```
PS C:\Users\nayak\OneDrive\Desktop\fim> C:\Users\nayak\OneDrive\Desktop\fim\FIM.ps1

What would you like to do?

    A) Collect new hashed?
    B) Begin monitoring files with saved hashed?

Please enter 'A' or 'B': b

C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
```

Fig.2

```
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test2.txt has changed!!!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test4 - Copy.txt has been created!
C:\Users\nayak\OneDrive\Desktop\fim\targetFiles\test3.txt has been deleted!
```

Fig.3

Conclusion

The CIA triad serves as a guiding framework for organizations to develop robust security policies. By addressing confidentiality, integrity, and availability, organizations can create a comprehensive security posture that protects sensitive data from various threats. In the context of file integrity monitoring, maintaining data integrity is particularly critical, as it directly impacts the trustworthiness of information and the overall security of the organization's data management practices.

Reference

- Youtube: Josh Madakor
- Fortinet: https://www.fortinet.com/resources/cyberglossary/cia-triad
- Coursera: https://www.coursera.org/articles/cia-triad