How to Verify Software File Integrity

The goal of software integrity verification is to ensure that what you receive is exactly what was sent from the software vendor and has not been modified. GE's Automation & Controls has been continuously improving our software integrity verification tools, first using *.hash* files, then product signing, and now expanding our signing capability and ease-of-use using PowerShell scripting. This document provides a history of improvement tools and the procedures to verify software file integrity using each tool.

History of Software File Integrity Verification Improvement Tools

Power Script Product Signing (Releases Dec 2015 and Later)

Beginning in December 2015, GE's Automation & Controls added a layer of PowerShell scripting to our deliveries. This provides a consistent interface to our integrity checking mechanisms and allows us to sign any file or a collection of files (rather than just files that are typically signed, such as *.exe* and *.msi* files). This applies to all deliveries, from full ControlST* release, to ControlST components, as well as collections of documentation and other products.

The basic procedure is as follows:

1. Verify the signature of the PowerShell script, *Verify-Integrity.ps1*, to confirm that the script itself has not been modified.
2. Run the script to verify all content and files associated with the script.

Refer to the section *PowerShell Script Product Signing File Integrity Verification Procedures (Releases Dec 2015 and Later)*.

Product Signing (Releases Sept 2015 through Dec 2015)

Beginning in September 2015, ControlST release deployment files (*Setup.exe*, **.msi*) have been signed by GE. This means GE's Automation & Controls has created an SHA-256 hash of the contents, and encrypted that hash using a Private Key. If you use GE's Automation & Controls' Public Key (also obtained through the certificate shipped with the file), then you know the file came from GE's Automation & Controls and has not been modified.

Refer to the section *Product Signing File Integrity Verification Procedures (Releases Sept 2015 through Dec 2015)*.

Hash File Checking (Releases Prior to Sept 2015)

Previously, GE's Automation & Controls created an SHA-256 hash of the deployment files and made that available *out of band* for checking against the file contents. There was no way to ensure that the hashes were as delivered by GE, other than the fact that you downloaded the files from GE's Automation & Controls's website.

Refer to the section *Hash File Integrity Verification Procedures (Releases Prior to Sept 2015)*.

Computer Requirements

Users are required to run Windows® 7 or later to use these instructions.

Note Windows operating systems prior to Windows 7 do not support the PowerShell feature.

PowerShell Script Product Signing File Integrity Verification Procedures (Releases Dec 2015 and Later)

Normal Operation

During normal operation, perform the following verification procedures:

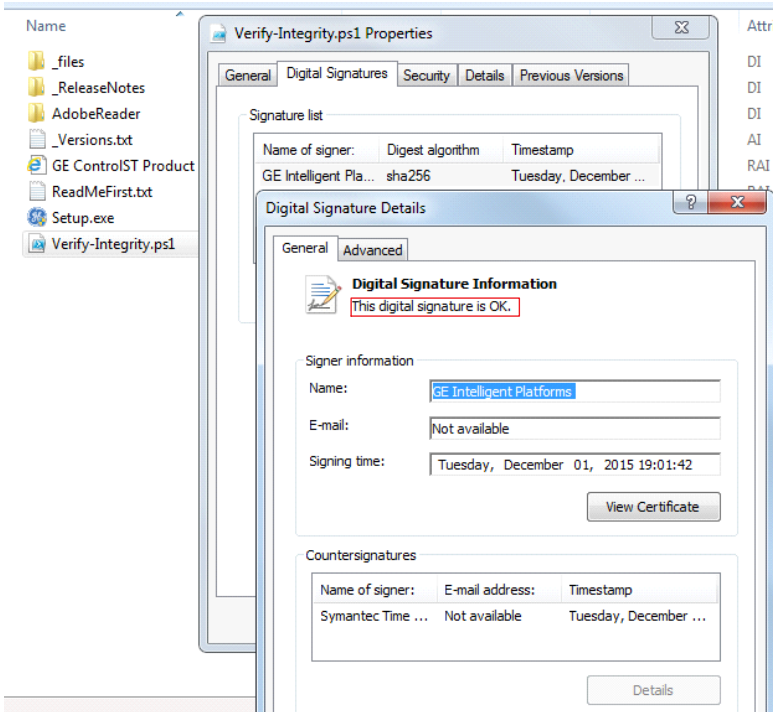
1. Verify that the *Verify-Integrity.ps1 script* is signed and trusted.
2. Open a Command prompt and enter the PowerShell console using the command PowerShell.
3. Set the execution policy using the command Set-ExecutionPolicy RemoteSigned –Scope CurrentUser.

Note GE Intelligent Platforms has been renamed as the Automation & Controls group within GE Energy Management; However, product signing will display *GE Intelligent Platforms* until we renew our Symantec certificate. This does not affect the integrity of the software.

Note Windows can run PowerShell scripts using the current Windows account. It verifies the product in its own directory and the user does not need to have the default directory be that directory.

➢ To verify the signature on the script

1. From Windows Explorer, navigate to the folder containing the *Verify-Integrity.psi* file.
2. Right-click the file and select **Properties**.
3. View the Digital Signature information and verify the digital signature is OK.

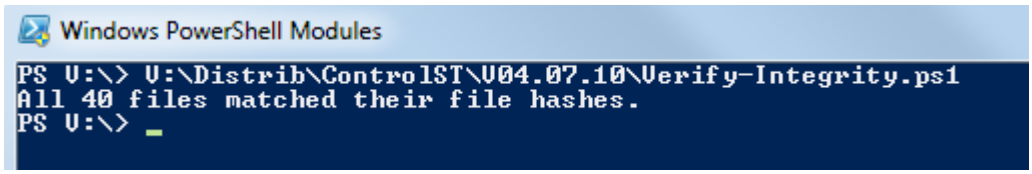


From the **Digital Signatures** tab, select the name of the signer (*GE Automation & Controls*) and click **Details**.

Verify that ***This digital signature is OK*** is displayed.

➢ **To verify the integrity of the files from the Windows PowerShell console**

1. Open a Command prompt and enter the PowerShell console using the command PowerShell.
2. Verify that all files match by running the *Verify-Integrity.ps1* script.



PowerShell Console Command Prompt File Integrity Verification

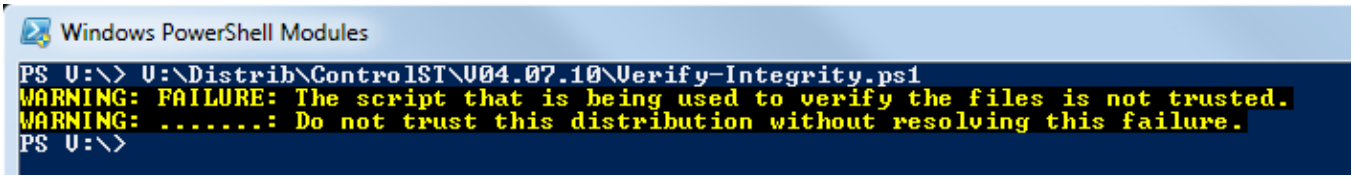
➢ **To verify the integrity of the files from the Windows (DOS) console**

1. Open a Command prompt and enter the DOS console using the command CMD.
2. Verify that all files match the by running the *Verify-Integrity.ps1* script.

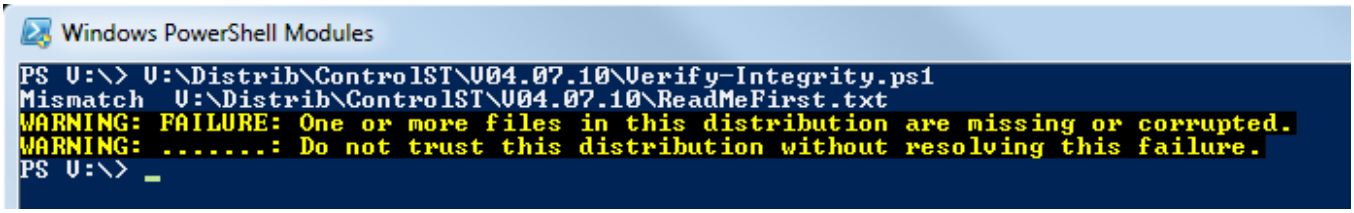
DOS Console Command Prompt File Integrity Verification

Troubleshooting

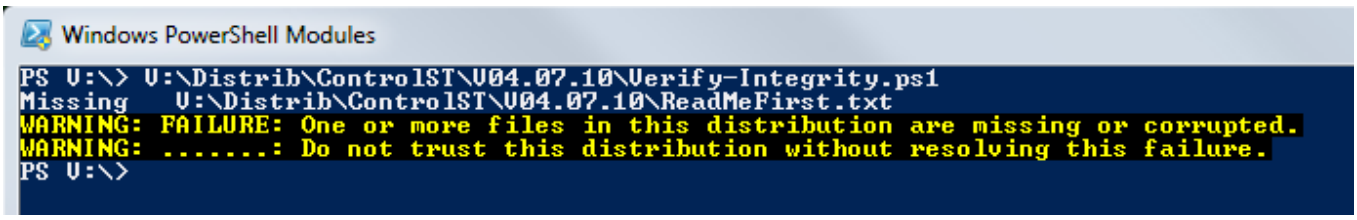
If any the following warnings/failures are displayed during PowerShell script file integrity verification, do not trust the file until the failure is resolved.



Verify-Integrity Script Integrity Check Failure Message



One or More Files Integrity Check Files Mismatch Message

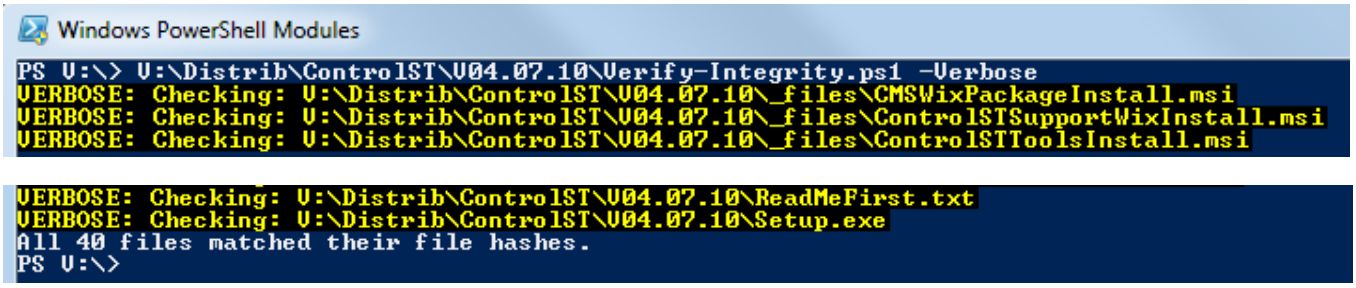


One or More Files Integrity Check Missing File Message

Verbose Option

➢ **To verify the integrity of the files using the Verbose option**

1. Open a Command prompt and enter the PowerShell console using the command Verbose.
2. Verify that all files match the by running the *Verify-Integrity.ps1* script.



Verbose Command File Integrity Verification

Product Signing File Integrity Verification Procedures (Releases Sept 2015 through Dec 2015)

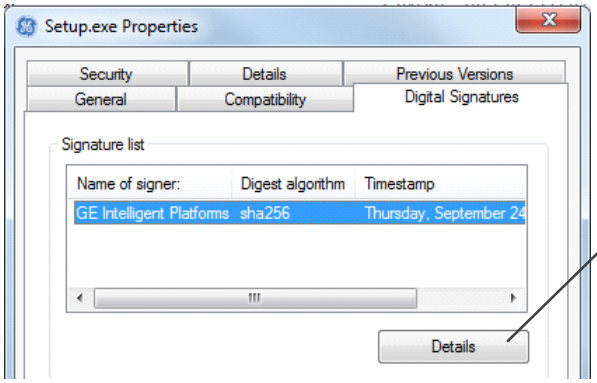
If a software release was signed with GE's Automation & Controls' external certificate, the certificate issuer is Symantec. If the release was signed with GE's Automation & Controls's internal certificate, then the certificate issuer is GE's Automation & Controls' internal domain (PDEV); you can only trust the file if you are a member of the PDEV domain or you have configured your computer to trust GE's Certificate Authority (CA) Server.

➢ **To verify the signature of a file in the ControlST release**

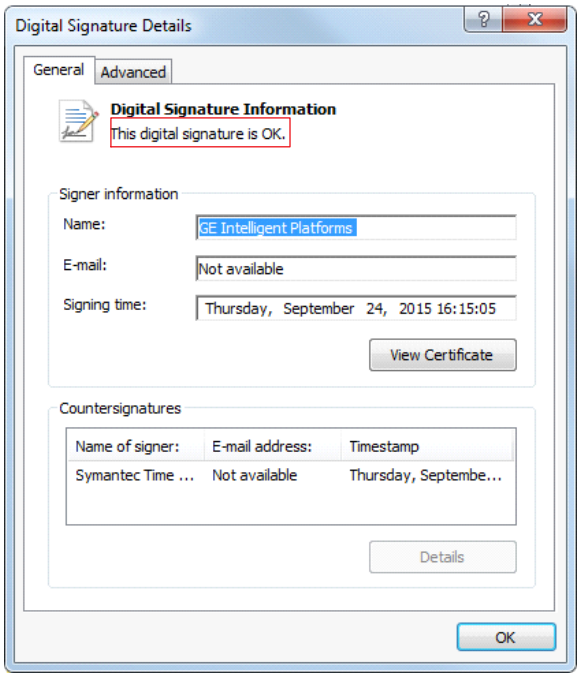
Note Only certain kinds of files can be signed, such as *.exe* and *.msi* files.

---

1. From Windows Explorer, navigate to the folder containing the file.
2. Right-click the file and select **Properties**.
3. View the Digital Signature information and verify the digital signature is OK.



From the **Digital Signatures** tab, select the name of the signer (***GE Automation & Controls***) and click **Details**.



Verify that ***This digital signature is OK*** is displayed.

Click **OK** to close the window.

This confirms the following:

- The hash of the file matches the hash value that was decrypted using GE's Automation & Controls' Public Key.
- The issuer (signer) of GE's certificate was trusted, therefore GE's Automation & Controls' certificate is trusted.
  - For GE's Public certificate, this indicates you trusted Symantec, Symantec trusted GE's Automation & Controls, therefore you trust GE's Automation & Controls.
  - For GE's Internal certificate, this indicates you trusted the PDEV domain CA, the PDEV domain CA trusts GE's Automation & Controls, therefore you trust GE's Automation & Controls.

As a convenience, GE's Automation & Controls has also added an additional level of integrity checking so the user does not have to verify every file in a ControlST release. The file Setup.exe verifies the integrity of the files that it launches, and requests user validation if it detects a file that is not trusted.

Attention

For self-extracting executable files: In addition to the previously discussed verification in this document, you may wish to verify the integrity of a self-extracting executable file that you downloaded from a GE website or file server. Within this executable, all files have been signed as previously described; However, the executable file itself may be too large to allow product signing. In these cases, a *.hash* file is used (available with the ControlST release). Follow the instructions provided in the section *Hash File Integrity Verification Procedures*.

Hash File Integrity Verification Procedures (Releases Prior to Sept 2015)

Note This procedure also applies to self-extracting executable files that are too large to be signed.

Acquire PowerShell Script

➢ **To acquire PowerShell script:** download the *Check-Filehash.ps1* PowerShell script from the following locations:

- Internal GE users: *\\pdevnt.salem.ge.com\Releases\ControlST\Other\Hash Checker*
- External GE users: contact *customer support*

Verify a Directory's Contents

Perform the following procedures to verify a directory's contents using a *.hash* file:

1. Acquire a previously generated *.hash* file.
2. Configure the PowerShell environment to grant Windows permission to run the PowerShell scripts defined by other users.
3. Verify the directory's contents using a single, previously generated *.hash* file.

➢ **To acquire previously generated *.hash* file:** Download the *.hash* file for the release.

Note External GE users must contact customer support to acquire the *.hash* file for the release.

GE Internal users can download the *.hash* file for the product or release for verification as follows:

- Internal GE Products (contains all full ControlST releases and component releases) are available on the *ControlST Releases File Server* . The *.hash* file is located within the folder that hosts the software download.
- External GE Products (contains General Market ControlST releases) are available on *GE's Automation & Controls Support website*.

➢ **To configure the PowerShell environment**

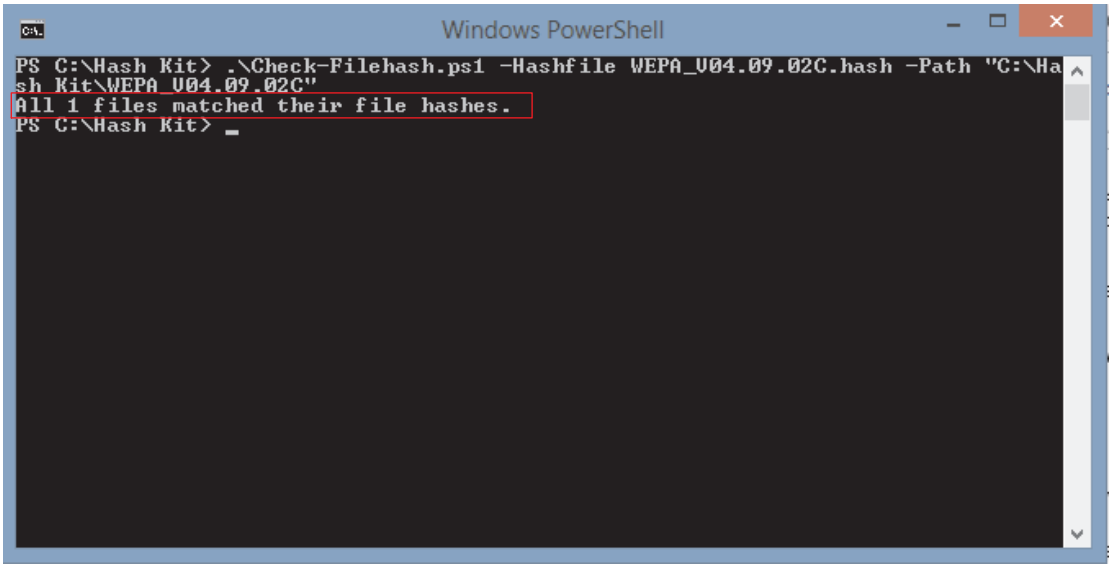
1. Open a Command prompt and enter the PowerShell console using the command PowerShell.
2. Set the execution policy using the command Set-ExecutionPolicy RemoteSigned –Scope CurrentUser.

Note Windows can run PowerShell scripts using the current Windows account.

➢ **To verify a directory's contents using a single, previously generated *.hash* file**

1. Open a Command prompt and enter the PowerShell console using the command PowerShell.
2. Check the directory contents against the *.hash* file using the command: .\Check-Filehash.ps1 –Hashfile HashfileName.hash –Path "Path of Directory to Hash".

The following figure illustrates an example output for checking a directory with the path C:\Hash Kit\WEPA_V04.09.02C against a previously generated file, *WEPA_V04.09.02C.hash*.



WEPA_V04.09.02C.hash File Integrity Verification Result (Passed)

Glossary of Terms

Key Pair: A cryptographic concept whereby two keys are generated such that if something is encrypted by one key it can only be decrypted by the other, and vice versa.

Public Key: The Key in the Key Pair that is made public or published.

Private Key: The Key in the Key Pair that is kept private.

Certificate: A Public Key along with the identity of the person who owns it, typically signed by a 3rd party you trust to vouch for the identity of the certificate holder (the person with the private key).

Hashing: The act of taking data of any size and running it through an algorithm to produce a single value of a fixed size. If the original data is changed the output value produced (hash value) also changes. The algorithm is chosen to make it extremely difficult to make changes in the original data while still having it generate the same output (hash) value.

Signing: The act of taking a hash of something (such as file, certificate, password) and encrypting the resulting hash with your Private Key. A receiver can then decrypt this hash with your Public key and check it against the contents of what was hashed. If they match, the recipient knows that it came from you (only you could sign it using your Private Key) and has not been modified (the hash values [one you computed, one you decrypted using my Public Key] match).

Help Option on Verify-Integrity.ps1 Script

The *help Verify-Integrity.ps1* command provides additional help and examples, such as follows:

PS V:\sign> help Verify-Integrity.ps1 -Detailed

NAME

Verify-Integrity.ps1

SYNOPSIS

Verify-Integrity.ps1 checks the integrity of the product located in the same directory as itself. Verify that the Verify-Integrity.ps1 file is signed then execute the Verify-Integrity.ps1 script to verify the integrity of the remaining files in the distribution.

SYNTAX

Verify-Integrity.ps1 [-Verbose] [CommonParameters]

DESCRIPTION

The Verify-Integrity.ps1 script is a signed script that will verify (using file hashes) the contents of a distribution directory. The script is signed to prove that it came from the manufacturer and has not been altered – verify the signature on the script file before running the script file.

The signature can be verified by opening Windows Explorer, navigating to the directory holding the distribution, right-clicking on the Verify-Integrity.ps1 file, and selecting "Properties" - "Digital Signatures". Select the line in the signature list and click on "Details". The resulting dialog should show the phrase "This digital signature is OK."

After verifying the signature on the Verify-Integrity.ps1 file, run the file to verify the integrity of the remaining files in the distribution. The script will check the files in the same directory as the Verify-Integrity.ps1 script, you do not need to set your current default directory there first. The script can be run from a PowerShell console, or from a Windows (DOS) console using the "PowerShell" command. (See the examples for more details.)

———————————— EXAMPLE 1 ————————————

PS C:\PS>:\Download\Verify-Integrity.ps1

The above command run from a PowerShell console verifies the distribution located in the D:\Download directory. (Make sure you verify the script file is signed before running it.)

———————————— EXAMPLE 2 ————————————

C:\>PowerShell -File D:\Download\Verify-Integrity.ps1

The above command run from a Windows console verifies the distribution located in the D:\Download directory. (Make sure to verify the script file is signed before running it.)