BLAST

TI FILE INTEGRITY

By Anthony Cagliano

Thank you for your interest in Blast: TI File Integrity Software. This program is a simple utility for ensuring the integrity of files on your TI graphing calculator. It implements a bunch of state-of-the-graphing-calculator-art, widely used and well-known algorithms to monitor the attributes of your TI programs and application variables. It can create snapshots of existing programs as well, which allow you to quickly and easily reset one to an earlier version.

In this manual, I will elaborate on the features and interface of this utility, and explain how every feature works. I will also inform on the memory usage and external storage used by the program for those who would like to know.

INSTALLATION

Install this program by sending it to the RAM of your TI-84+ CE graphing calculator. This program will not work on any other model of calculator. If you have a shell such as Cesium or DoorsCE with run-from-archive capabilities, you can send this program to the Archive of your device instead. In addition, you will need to have the C libraries by MateoConLechuga (link will be in the *Dependencies* section towards the end of this manual), as this program makes use of its graphics and file-stream functions. If you do not have these installed, the program will terminate, printing an error referencing the libs, and a tiny.cc web URL that links to the libs. This will also occur if the libs are out of date versus the lib versions that the utility was compiled with.

This utility will install a few other files for its own use… a file attributes database, a settings file, and a number of snapshot files (if snapshots are created). In addition, you can optionally send the malware definitions file to your calculator, an application variable by the name of *AVMALDEF*. Not having this installed will not stop you from using the file integrity/snapshot features of this program, but having it installed allows you to scan files, as well as to use the silent *System Time restore* feature that is built in to the program.

USAGE

When you first run this program, it will create 2 empty files… *AVPropDB* and *AVSett*. These are application variables and they will contain the file attributes database and the program settings, respectively. It will then begin the indexing process. The first time you run this program, the indexing process could take a while (upwards of several minutes). This is because it is reading the names, types, sizes, and compiling checksums of every protected program on your calculator. We limited the default focus of this program to protected programs because of their ability to execute code you cannot see. You can expand this to

include application variables and normal unlocked programs if you wish to look at more than the default. Bear in mind that you cannot checksum or snapshot an un-indexed file, so if you, for example, want to snapshot a TI-Basic program you are working on, you will need to allow this program to index normal programs and then re-index. Before we talk about the individual functions of the utility, let's talk about the Settings.

SETTINGS

1) Enable Split Indexing Mode – This is an option that is meant to shorten the extended indexing time for people who have lots of stuff on their calculator. This causes the on-load index to skip the checksum, which is the most time-consuming part of the process. The checksum will instead be calculated when you attempt to view the program's attributes in the [File Options] menu.

2) Index Unlocked Programs – This is an option that allows you to track and snapshot unlocked, unprotected programs including TI-Basic programs. These are excluded by default because programs of this type are usually open to having their source code viewed.

3) Index Application Variables – This is an option that allows you to track and snapshot application variables (appvars). These are excluded by default because application variables are typically used to store data, and it is very easy to obfuscate or compress data, and their sizes and content are always changing, making monitoring or scanning an appvar unpractical. Same as with Unlocked programs, we understand there are times you may want to monitor or scan an appvar, and so we allow you to change this default behavior.

4) Snapshot Maximum Slots – This setting restricts the maximum number of snapshot files you are allowed. Since snapshots for the same file are saved into the same snapshot file, this essentially limits the number of files you can keep snapshots **of**, not the number of snapshots in total.

FILE OPTIONS

File Options is the main, often go-to menu in this utility, as it is the path many of the other functions. From within this menu you can see the name, type, size, and checksum of every indexed file on your device. You will also see a small green or red icon next to size and checksum if attributes tracking is enabled for that file, depending on whether or not the current attributes match the recorded ones. You can also see an abridged note on whether or not you have a snapshot for this file. You will see *Enabled* if a

snapshot for that file is found, and *None* otherwise. By pressing [Enter] from within this menu, you can bring up a select bar to perform actions on the file you are currently viewing. You can choose to *Enable Attribute Tracking* on the file, which saves its name, type, size, and checksum to the attributes database (and changes this menu option to *Disable Attribute Tracking*). You can choose to *Update Attributes*, which overwrites the older save of the attributes to the current ones in the database. Lastly, you can choose to *Scan File* which loads the malware definitions (if present) and looks for matches in the current file. If a match is found, you are told the address in memory at which it occurs. If these options occur successfully, then you will be presented with a green checkmark next to the option. If these options fail, you will be presented with a red X. In the subsection titled "Errors", I will go over the possible error conditions that can be returned.

Lastly, you can Enable or Update a snapshot from within this menu. Enabling snapshots on a file saves the current contents of a file to a snapshot which can be reverted to, even if the source file is deleted from your device. Every snapshot saved also has a timestamp which is used to differentiate between snapshot versions. Once a snapshot exists, the menu option switches to *Update Snapshot*, which allows you to record another newer version of the file. Snapshots for the same file (name and type match) are saved into the same snapshot file.

Since indexing is key to the functioning of this program, sometimes (for example after a Settings change) you may need to re-index your device. You can do this by pressing the [Stat] key from within this or any menu.

SYSTEM SCANS

This is one of the new features that sets this version of BLAST apart from its predecessor. This program has the ability to return and track the attributes of the operating system itself. Due to the time it takes to return a checksum for the OS, this is not done during program load—you must request a checksum of the OS by selecting the *Checksum OS* menu option. After about a minute, the current checksum of the OS will be displayed. Simultaneously, the size and checksum values will gain a green or red icon next to them if an OS checksum has been recorded.

Once an OS checksum has been returned, you will be able to save the checksum and size of the OS to the attributes database, so that you can also monitor it for changes. Lastly you can also *Scan OS*, which scans the OS itself for matches to the malware definitions. If a match occurs, you are told the address in memory at which this match occurs. You are unable to snapshot the OS, for obvious reasons.

SNAPSHOT

This is perhaps one of the more useful features of this program, considering that file integrity and malware scanning have no real purpose on a calculator at this time. The snapshot feature allows you to record the state of a file into memory which can then be restored later if needed. Timestamps differentiate between different snapshot versions and you can revert to any of them. Through the Snapshot menu you can see the snapshots recorded for files, all versions saved, their timestamps, and modify/delete those snapshots. By scrolling through the timestamps and pressing [Enter] you can revert to the presently selected snapshot version. By selecting *Delete Snapshot*, you delete the entire snapshot file. At present, you cannot delete only one version of a snapshot.

Because snapshots are copies of a file and are not compressed (yet), the snapshot feature is a memory-heavy feature. It helps that the snapshot files are dumped into archive after creation, but they still add up. The Settings option for Snapshot number limit serves to limit the number of snapshots you can create, but since you can raise this option as high as you want you will have to track your remaining memory. Attempting to save a snapshot with not enough memory will cause the snapshot creation to fail.

ERRORS

While this utility has very few error conditions, there are a few that you should be aware of. It is possible for the Enable/Disable of Attribute Tracking on a file or the OS to fail, usually because of inadequate memory to complete the operation… or more accurately, failure to resize the attributes file. It is also possible for the creation, updating, and reversion of a snapshot file to fail for the same reasons. This will cause the program to display an error icon (a red circle with a white X in the middle) next to the function you just tried to perform. If the operation succeeds you will be shown a success icon (a green circle with a while checkmark).

Last but not least, the malware scanning can fail if the malware definitions file is not present. This also causes the aforementioned error icon to show up next to the *Scan* menu option you just tried to run. This program has fairly good error-catching that I so far have not found a bug in.

TECHNICAL INFORMATION

This program has intricate internal functionality, made possible by a number of configurations, both internal and external. The removal of any of these features may not break the program, but might render it unable to adequately perform the tasks it is supposed to. In this section I will detail the memory and files this program needs and what their removal might do.

DEPENDENCIES

This program requires the C libs by MateoConLechuga to function. You may find these at https://github.com/CE-Programming/libraries.

You may also find the Malware Definitions file, *AVMALDEV*, useful for enabling the scanning functions of this utility. It can be found at http://clrhome.org/blastav/downloads.php.

FILES USED BY PROGRAM

AVMALDEF, TI Application Variable.
- Optional, downloaded and installed by user.
- Size unknown, depends on number and length of malware definitions.
- File serves as database of malware to search for.
- Removal does not affect program runtime.

AVSETT, TI Application Variable.
- Necessary, created by program on first load or if not found.
- Size: 3 bytes at present
- Stores program runtime settings.
- Removal resets program settings.

AVPropDB, TI Application Variable.
- Necessary, created by program on first load or if not found.
- Size: 16 bytes per file tracked
- Stores attribute tracking for files (or OS).
- Removal deletes all attribute saves.

AVshXXXX, TI Application Variable.
- Optional, created by program for snapshot saves.
- XXXX is an index number (Ex: AVsh0001 for snapshot #1)

Size: variable, depends on the size of the saved snapshot versions saved. Removal deletes the saved snapshots in that file.

RAM USED BY PROGRAM

This program makes use of a significant amount of RAM to conduct its functions. Excluding the placement of the above listed files into RAM for reading during the course of the program's runtime, the following blocks of memory are used by the program:

1) A block of dynamically-allocated memory to hold the index of all files on your device. The size of this block is 17 bytes per indexed file.
2) A block of dynamically-allocated memory to hold the index of all snapshots on your device. The size of this block is 19 bytes per indexed snapshot.
3) One statically-allocated settings instance, 3 bytes large.
4) One statically-allocated OS attributes instance, 6 bytes large.
5) One statically-allocated selected structure, 13 bytes large.
6) 7 dynamically-allocated sprites, totaling 4400 bytes.
7) ~50 bytes statically-allocated scratch RAM

Please ensure that you have the adequate memory available to run this program. While I took care to wrap all memory-required functions in error handlers, there is always the chance that running this program in a low-RAM setting might affect some things.

LEGAL

I, Anthony Cagliano, do hereby release and save harmless myself from any and all liability that arises from the usage of this utility. While great care has been taken to ensure stability in all aspects of this software, please understand that you use it at your own risk. Please also understand that using this software in a manner inconsistent with the directives laid out in this document can lead to numerous unintended results, including but not limited to data corruption or loss, NMI Resets (Crashes), or the bricking of the device. While these are highly unlikely even in extreme circumstances, they are a distinct possibility.

Due to the nature of this software, this program is closed-source. The source code to this project is restricted to developers, and the program itself is compressed. While you may freely share the software and this document with another calculator user, it is not permissible to reverse engineer or modify this program or any of its peripheral files, and it is certainly not permissible to distribute modifications without my express permission.