PurgeTMP

Purge unopened temporary disk files

Protection against a Denial of Service attack, caused by temporary disk files

23. January 2006

Dear GreenHouse tool user,

while working on a manual describing methods to harden your NonStop server, I stumbled into the need for a tool, that keeps my volumes clean from unopened temporary disk files.

What are temporary files?
Temporary files are files you can' t see the easy way: They don't reside in your default location, nor in any other location (subvol): A temporary disk file resides on a disk volume.

A temporary file can be created by anybody, using the CREATE command of TACL, or FUP, or programmatically by user the proper GUARDIAN procedure call.

The command syntax is:

```
CREATE <$vol>
```

To create a temporary file on $GHS1, simple execute this command from e.g. your TACL:

```
CREATE $GHS1
```

The command creates a new file on $GHS1, where the file name is chosen by the system.
The name format is: `$vol.#nnnnnnn`, where:

`$vol`	is the volume name, on which the file resides
`#nnnnnnn`	is the file name, starting with the hash sign, and followed by a 7 digit decimal number. The number is in the range of 0000000 .. 9999999

The danger is, that you can repeat this CREATE command WITHOUT receiving an error 10 (File already exists). So any repeated CREATE causes a new entry in the disk directory.

Temporary disk files do NOT reside in a location (subvol). To find them, you have to perform one of these commands:

- `FILEINFO $vol.#*` or
- `FUP INFO $vol.#*`

The following example shows a FUP command, and its result:

```
$GHS1 PURGETMP 139> fup info $ghs1.#*
              CODE              EOF      LAST MODIF    OWNER RWEP    TYPE    REC BL
$GHS1.#0000000
          O    440                0           09:10 100,5    UUOO

$GHS1.#0000005
          O    440                0           09:10 100,5    UUOO

$GHS1.#0000006
          O                       0           09:10     -1    OOOO

$GHS1.#0000007
          O    440                0           09:10 100,5    UUOO

$GHS1.#0000037
                 0                0           19:10 100,5    UUOO
```

All temporary files of the volume in question are displayed.
In the example above, all temporary files are open, except file #0000037: This file is unopened yet!

What happens to file #00000037?
It stays on $GHS1, until it is purged explicitly, or it is opened and later on closed, which automatically initiates a system triggered purge.

What are temporary files good for?
Temporary files are pretty neat: When you need a file, just create on, open it, and use it. And when you are done, simply close the file, and the system performs an automatic purge, when the last open is closed. This means: You don't need to take care of temporary files.

No big deal – so far.

Why might temporary files be a problem?
The main problem is, that the creation of temporary files can not be controlled and denied by the security mechanism (GUARDIAN, SAFEGUARD). Every user with interactive access to the system (e.g. through TACL) can create temporary files everywhere on the system. Even volume ACLs don't stop this. This is in contrast to 'normal' files: This type of file creation can be completely controlled, and denied when necessary.

Let us assume, that a piece of software has a bug, and performs a recursive create of a temporary file, or even worse: There is someone doing this by intention, using a small TACL Macro, or program. What happens? Creating, and not purging, files for a long time for sure fills up the disk file directory, and when it runs full, the disk volume in question has a massive problem: It simply refuses from working any longer, and optionally goes down. And in case the volume is $SYSTEM, the system may follow the disk...
This type of attack can be done even through EXPAND: All, the attacker needs, is remote passwords...

Pretty scary, isn't it?

What to do?
Best would be to have security measures in place, controlling temporary files. But there are none. So next best would be, to have a tool, that scans the volumes of your system on a regular basis, and automatically purges all unopened temporary files, which are older than a given time, e.g. 10 seconds. Normally a temporary file is created, and immediately opened, by the creator. Creating a temporary file, and NOT opening it within e.g. 10 seconds may point to a problem... (or attack?).

Based on this I invested a few days and developed a new GreenHouse FreeWare tool, named: PURGETMP.

PURGETMP performs an automatic clean-up of unopened temporary disk files.

To make it at efficient as possible, the following action attributes can be supplied to the program at start time:

VOLUME	Defines the volume(s) to be checked. VOLUME allows extended wildcards, such as: $GHS{..n} In case no VOLUME is supplied, all available volumes are checked. The default is: $* The maximum number of disk volumes that can be checked by one PURGETMP process is 30,000 (thirtythousand).
DELAY	The DELAY value defines in seconds, how often the PURGETMP tool should run. Valid values are: 1 .. 3,600 seconds In case the entry is missing, 30 seconds are used as default delay.
FILEAGE	As explained above: A temporary file is created, and immediately opened by the creator. But there still is some time left between the creation and subsequent open. To prevent PURGETMP from purging a just created und not yet opened temporary files, a minimum file age can be defined: Unopened temporary file, younger than the given file age, are NOT touched. Valid file age values are: 0 .. 300 seconds In case the entry is missing, a file age of 30 seconds is assumed.
REFRESH	This number defined the number of check rounds, before PURGETMP performs an inventory run: It checks again for the available volumes, and re-computes the directory offsets. A REFRESH is done as well when the number in the name of a temporary file is bigger than 99,999 (e.g. $SYSTEM.#9999925) Valid values are: 1 .. 1,000 In case the entry is missing, a value if 100 is assumed.
THRESHOLD	Minimum number of purged files, before PURGETMP creates a critical EMS message.

When PURGETMP has to purge a lot of files there is a pretty high chance, that something is going wrong on your system. PURGETMP takes care of this situation, and automatically adjusts the DELAY time in relation to the number of purged files:

- When 1 .. 99 files are purged, PURGETMP goes to sleep for the configured DELAY time
- When 100 .. 999 files are purged, PURGETMP goes to sleep for a 10th of the configured DELAY file
- When 1,000 or more files are purged, PUTGETMP starts over again immediately.

Multiple copies of PURGETMP can be executed, running in different CPUs and checking different volumes. This allows a fine tuning in regard to performance.

What happens to open temporary files?
Nothing: They stay alive until the last OPEN is closed.
When this is a problem, there is only one way in enforcing the purge: By stopping those processes, having an OPEN on that file.

May be I enhance PURGETMP with the function to kill all processes, having more than a configured number of temporary files open. e.g. when a process has more than 30 temporary files open, it gets stopped – because this is not normal.

Installation
PURGETMP comes in two flavours:

1. As NON native object (file code 100)
2. As native object (file code 700)

Use the object that best performs on your system.

The security setting should be:
Owner: SUPER.SUPER, or System Administrator
Security: "OOAO", or an equivalent SAFEGUARD ACL

The program does not contain PRIV code, thus does not need to become licensed.
Command syntax is:

Program Execution
To allow PURGETMP to purge ANY unopened temporary file, independent of the owner, it has to be started from the SUPER.SUPER ID.

```
[run] PURGETMP [/…/] [attribute]
```

where `attribute` is one or more of:

```
VOLUME    <$vol>
DELAY     <seconds>
FILEAGE   <seconds>
REFRESH   <num>
THESHOLD <num>
```

e.g.

```
$ghs1.purgetmp.purgetmp/name $clean,pri 199/threshold 2,delay 5,volume $GHS*
```

This command starts PURGETMP at a priority of 199 with the name $CLEAN for all disk volumes, matching the template $GHS, by using the defined values for THRESHOLD, and DELAY. The attributes FILEAGE and REFRESH are used by their default values.

The start-up attribute NOWAIT is not necessary: PURGETMP wakes the TACL it is started from.
When PURGETMP is started, it displays its configuration to the OUT file.
In case OUT does not exists, it becomes created as an EDIT type file; in case it does exist, the output is written to the existing file.

A typical start looks like this:

```
$GHS1 PURGETMP 89> purgetmp/name $clean,pri 199/
PURGETMP (101) - T7172G06 - (03Jun2004)   System \BEECH, running NSK G06
Copyright (c) GreenHouse Software & Consulting 2004
Used VOLUME name/template: $*
Effective volume(s):       $SYSTEM
                           $GHS2
                           $GHS1
                           $DSMSCM
File age:                  30 seconds
Delay time:                30 seconds
Threshold:                 10 purged files
Num rounds before refresh: 100
$GHS1 PURGETMP 90>
```

Event Message Generation

Every file purge is reported to the EMS system:

- The purge of files of up to the threshold number generates an informational message.
- The purge of files equal to or bigger than the threshold number generates a critical message.

An EMS message looks like this:

```
04-06-03 13:02:49 \BEECH.$CLEAN     GHS.20.101            007172 Volume $DSMSCM
                                    cleaned from 1 unopened temporary file
```

Something to keep in mind

PURGETMP does not have any error handling for the case, that a temporary file can't be purged. This happens, when PURGETMP does not run with the SUPER.SUPER ID.
In other words: Executing PURGETMP with an ID other than SUPER.SUPER may result in the fact, that no unopened temporary files are purged.

How many temporary files have to be expected?

Zero!
In a normal environment, no unopened temporary files are encountered.
Finding unopened temporary files is caused by:

- A bug in one of your programs
- Intention: Someone is hacking your system

What is the best method of starting and continuously running PURGETMP?
The idea to start PURGETMP through CIIN at cold load time is not the best one. CIIN should be kept as clean as possible!
A better place to start PURGETMP from is in your system starts file: First start the subsystems, such as TMF, SPOOLER etc., then start PURGETMP.
Another perfect place is the use of the $ZZKRN system. This method is available with the S machines (G releases). Starting something through $ZZKRN ensures, that the program in question is automatically started at system cold load time, and automatically re-started in case it stopped (for what reason ever).

To configure PURGETMP in $ZZKRN:

1. Edit the TACL Macro below, and change it according to your needs.
2. Logon to SUPER.SUPER, and execute it.
   It adds the PURGETMP tool as a permanent service to your system, and starts it.
   From now on, PURGETMP is automatically started when your system is started.

```
?TACL Macro
#FRAME
#PUSH #INLINEPROCESS

inlprefix -
scf/inline/


- allow all errors
- assume process $zzkrn

- abort  #purgetmp
- delete #purgetmp

- add #purgetmp &
, autorestart 10 &
, cpu first &
, highpin on &
, name $CLEAN &
, outfile $0 &
, priority 199 &
, program $ghs1.purgetmp.PURGETMP &
, startmode application &
, startupmsg "VOLUME $*, DELAY 30, FILEAGE 30, REFRESH 1000" &
, userid 255,255

- start  #purgetmp
- info   #purgetmp,detail
- status #purgetmp
- exit

#POP #INLINEPROCESS
#UNFRAME
```

Please adjust the yellow/grey marked commands according to your needs!

Performance implications
PURGETMP is developed with the goal, to create a perfect product:

- Effective (it does what it claims to do)
- Fast (no unnecessary CPU cycles)
- Self optimizing (it adapts itself constantly to the changing environment)

This goal – by the way – is base for ALL GreenHouse

- FreeWare products
- ShareWare products and
- Commercial products

Does PURGETMP solve all DoS problems?
Of course NOT! But it makes your system much more secure against a DoS attacks, that is based on the creation of temporary disk files.

How much is PURGETMP?
Zero - of course!
In numbers: 0.00 US$ (for the non Europeans), which is 0.00 € (for us Europeans).

PURGETMP is FreeWare!

GreenHouse does not charge for every service, and product, it provides.
As a long time Tandemite (first contacts in 1976, employed by the company starting 1978 until 1994, Alliance Partner since then) we are interested in having a satisfied customer and tool user base which also helps Tandem to sell their products. When Tandem sells products to happy customers, GreenHouse sells products as well!

Fell free to use and distribute PURGETMP!

Related tool
GreenHouse has another tool, that protects your system from a DoS attack: The ShareWare tool ListLib prevents the listner from flooding your system with unauthorized FTPSERV processes. For more details, please check: http://www.greenhouse.de/shareware/ListLib.html

Related Product
The product PrCoSEEP keeps track of the type, and number, of processes, a user is allowed to execute simultaneously. Please check: http://www.greenhouse.de/products.php

In case you stumble into problems, or questions: Please do not hesitate to contact me any time at: Info@GreenHouse.de

greenHouse
Software & Consulting
www.GreenHouse.de