Suricata - Bug #2433
memleak with suppression rules defined in threshold.conf

02/04/2018 08:54 AM - Peter Manev

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>Carl Smith</td>
</tr>
<tr>
<td>Category:</td>
<td></td>
</tr>
<tr>
<td>Target version:</td>
<td>6.0.0rc1</td>
</tr>
<tr>
<td>Affected Versions:</td>
<td></td>
</tr>
<tr>
<td>Difficulty:</td>
<td></td>
</tr>
<tr>
<td>Effort:</td>
<td></td>
</tr>
<tr>
<td>Label:</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

If there is a suppression for a rule in the threshold.conf - for example -

```plaintext
suppress gen_id 1, sig_id 2011813, track by_src, ip 10.0.0.0/16
```

valgrind reports memleak -

```
[1810] 4/2/2018 -- 15:04:37 - (detect-engine-build.c:1704) <Info> (SigAddressCleanupStage1) -- cleaning up signature grouping structure... complete
==1810==
==1810== HEAP SUMMARY:
==1810== in use at exit: 19,935 bytes in 386 blocks
==1810== total heap usage: 665,243 allocs, 664,857 frees, 105,359,022 bytes allocated
==1810== 7 bytes in 1 blocks are definitely lost in loss record 26 of 381
==1810== at 0x4C2BBAF: malloc (vg_replace_malloc.c:299)
==1810== by 0x6AFEA69: pcre_get_substring (pcre_get.c:569)
==1810== by 0x3C649B: ParseThresholdRule (util-threshold-config.c:819)
==1810== by 0x3C79D2: SCThresholdConfAddThresholdtype (util-threshold-config.c:1015)
==1810== by 0x3C7C5E: SCThresholdConfParseFile (util-threshold-config.c:1126)
==1810== by 0x3C1774: SCThresholdConfInitContext (util-threshold-config.c:219)
==1810== by 0x1FE964: SigLoadSignatures (detect-engine-loader.c:363)
==1810== by 0x3A976: LoadSignatures (suricata.c:2373)
==1810== by 0x32B307: PostConfLoadedDetectSetup (suricata.c:2504)
==1810== by 0x32C79D: main (suricata.c:2851)
==1810==
```

```plaintext
{<insert_a_suppression_name_here>
Memcheck:Leak
match-leak-kinds: definite
fun:malloc
fun:pcre_get_substring
fun:ParseThresholdRule
fun:SCThresholdConfAddThresholdtype
fun:SCThresholdConfParseFile
fun:SCThresholdConfInitContext
fun:SigLoadSignatures
fun:LoadSignatures
fun:PostConfLoadedDetectSetup
fun:main
}
```

```plaintext
==1810== 12 bytes in 1 blocks are definitely lost in loss record 35 of 381
==1810== at 0x4C2BBAF: malloc (vg_replace_malloc.c:299)
==1810== by 0x6AFEA69: pcre_get_substring (pcre_get.c:569)
==1810== by 0x3C65C7: ParseThresholdRule (util-threshold-config.c:825)
==1810== by 0x3C79D2: SCThresholdConfAddThresholdtype (util-threshold-config.c:1015)
==1810== by 0x3C7C5E: SCThresholdConfParseFile (util-threshold-config.c:1126)
==1810== by 0x3C1774: SCThresholdConfInitContext (util-threshold-config.c:219)
==1810== by 0x1FE964: SigLoadSignatures (detect-engine-loader.c:363)
```
==1810== by 0x32A976: LoadSignatures (suricata.c:2373)
==1810== by 0x32B307: PostConfLoadedDetectSetup (suricata.c:2504)
==1810== by 0x32C79D: main (suricata.c:2851)
{
    <insert_a_suppression_name_here>
    Memcheck:Leak
    match-leak-kinds: definite
    fun:malloc
    fun:pcre_get_substring
    fun:ParseThresholdRule
    fun:SCThresholdConfAddThresholdtype
    fun:SCThresholdConfParseFile
    fun:SCThresholdConfInitContext
    fun:SigLoadSignatures
    fun:LoadSignatures
    fun:PostConfLoadedDetectSetup
    fun:main
}
==1810== LEAK SUMMARY:
==1810==    definitely lost: 19 bytes in 2 blocks
==1810==    indirectly lost: 0 bytes in 0 blocks
==1810==    possibly lost: 0 bytes in 0 blocks
==1810==    still reachable: 19,916 bytes in 384 blocks
==1810==    suppressed: 0 bytes in 0 blocks
==1810== Reachable blocks (those to which a pointer was found) are not shown.
==1810== To see them, rerun with: --leak-check=full --show-leak-kinds=all
==1810==
==1810== For counts of detected and suppressed errors, rerun with: -v
==1810== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)

Using -


pevma@DONPEDRO:~/Work/Suricata/suricomp/tests$ suricata --build-info
This is Suricata version 4.1.0-dev (rev d2121945)
Features: PCAP_SET_BUFF AF_PACKET HAVE_PACKET_FANOUT LIBCAP_NG LIBNET1.1 HAVE_HTP_URI_NORMALIZE_HOSTS PCRE_JIT HAVE_NSS HAVE_LUA HAVE_LUAJIT HAVE_LIBJANSSON TLS MAGIC
SIMD support: SSE_4_2 SSE_4_1 SSE_3
Atomic intrinsics: 1 2 4 8 16 byte(s)
64-bits, Little-endian architecture
GCC version 6.3.0 20170516, C version 199901
compiled with _FORTIFY_SOURCE=0
L1 cache line size (CLS)=64
thread local storage method: __thread
compiled with LibHTP v0.5.25, linked against LibHTP v0.5.25

Suricata Configuration:
AF_PACKET support: yes
PF_RING support: no
NFQueue support: no
NFLOG support: no
IPFW support: no
Netmap support: no
DAG enabled: no
Napatech enabled: no
Unix socket enabled: yes
Detection enabled: yes
Libmagic support: yes
libnss support: yes
libnspr support: yes
libjanssons support: yes
liblzma support: no
hiredis support: no
hiredis async with libevent: no
Prelude support: no
PCRE jit: yes
LUA support: yes, through luajit
libluajit: yes
libgeoip: yes
Non-bundled htp: no
Old barnyard2 support: no
Hyperscan support: no
Libnet support: yes
Rust support (experimental): no
Experimental Rust parsers: no
Rust strict mode: no
Rust debug mode: no
Suricatasc install: yes
Profiling enabled: no
Profiling locks enabled: no

Development settings:
Coccinelle / spatch: yes
Unit tests enabled: no
Debug output enabled: no
Debug validation enabled: no

Generic build parameters:
Installation prefix: /usr
Configuration directory: /etc/suricata/
Log directory: /var/log/suricata/

--prefix /usr
--sysconfdir /etc
--localstatedir /var

Host: x86_64-pc-linux-gnu
Compiler: gcc (exec name) / gcc (real)
GCC Protect enabled: no
GCC march native enabled: yes
GCC Profile enabled: no
Position Independent Executable enabled: no
CFLAGS -ggdb -00 -march=native
PCAP_CFLAGS -fI/usr/include
SECCFLAGS

History

#1 - 02/09/2018 02:55 AM - Victor Julien
  - Status changed from New to Assigned
  - Assignee set to Richard Sailer
  - Target version set to 70

Richard, do you want to check this one out?

#2 - 02/18/2019 10:15 PM - Andreas Herz
  - Assignee changed from Richard Sailer to OISF Dev

#3 - 08/07/2020 02:05 PM - Victor Julien
  - Target version changed from 70 to TBD

Are these still valid?

07/16/2022
Victor Julien wrote in #note-3:

Are these still valid?

From code inspection - yes.

Bottom of util-threshold-config.c: ParseThresholdRule only frees th_track, th_count, th_seconds and th_type in the error case.

Something like this should fix it.

diff --git a/src/util-threshold-config.c b/src/util-threshold-config.c
index 2e5977841..b42f41aef 100644
--- a/src/util-threshold-config.c
+++ b/src/util-threshold-config.c
@@ -699,7 +699,7 @@ static int ParseThresholdRule(DetectEngineCtx *de_ctx, char *rawstr,
  *retParsedSeconds = parsed_seconds;
  *retParsedTimeout = parsed_timeout;
  *retThIp = th_ip;
+    th_ip = NULL;
+    res = 0;
error:
  if (th_track != NULL)
    SCFree((char *)th_track);
@@ -970,7 +970,7 @@ error:
    SCFree((char *)th_ip);
  - return -1;
  + return res;
} 
/**

#6 - 08/26/2020 05:57 AM - Victor Julien

- Status changed from Assigned to In Review
- Assignee changed from OISF Dev to Carl Smith
- Target version changed from TBD to 6.0.0rc1

https://github.com/OISF/suricata/pull/5310

#7 - 08/26/2020 05:58 AM - Victor Julien

- Status changed from In Review to Closed

Fixed by pr 5310 plus an additional fix https://github.com/OISF/suricata/pull/5325/commits/d3cf2c21df625de9d9dcd605f110e3bf76e5601