tianocore

Reporting Issues

Bug 1816 (CVE-2021-28211) - Possible heap corruption with LzmaUefiDecompressGetInfo

Status: RESOLVED FIXED

Alias: CVE-2021-28211

Product: Tianocore Security Issues
Component: Security Issue (show other bugs)
Version: Current
Hardware: All All

Importance: Normal normal
Assignee: Laszlo Ersek

URL:
Keywords:

Depends on:
Blocks:

Reported: 2019-05-15 13:55 UTC by Satoshi Tanda
Modified: 2021-09-22 15:47 UTC (History)
CC List: 16 users (show)

See Also:
Release(s) the issue is observed: edk2-stable202008
The OS the target platform is running: ---
Package: IntelFrameworkModulePkg, MdeModulePkg
Release(s) the issues must be fixed: edk2-stable202011

Attachments	
[PATCH] MdeModulePkg/LzmaCustomDecompressLib: catch 4GB+ uncompressed buffer sizes (3.85 KB, patch) 2020-09-28 12:22 UTC, Laszlo Ersek	Details | Diff
CVE .json file_2 (907 bytes, application/json) 2021-03-04 14:29 UTC, kevinj	Details
Add an attachment (proposed patch, testcase, etc.)	Show Obsolete (1)

Note
You need to log in before you can comment on or make changes to this bug.

Satoshi Tanda 2019-05-15 13:55:17 UTC Description

```
The LzmaUefiDecompressGetInfo function fetches 8 bytes (64bit-wide) of the
uncompressed data size from the LZMA header then silently drops the upper 32bit by
casting it to UINT32.

https://github.com/tianocore/edk2/blob/master/MdeModulePkg/Library/LzmaCustomDecompressLib/LzmaDecompress.c#L146

Since the caller of the function typically allocates heap based on the reported
size by this function to subsequently decompress data, the caller may allocate
smaller buffer than necessary when the input LZMA headers is greater than 32bit-
wide. For example, when the uncompressed size claimed in the header is
0x1_00000100, the LzmaUefiDecompressGetInfo function reports 0x100 to the caller.
The caller would allocate that heap and call the LzmaUefiDecompress function.
Because the LzmaUefiDecompress function fetches an uncompressed size from the
header again which will be 0x1_00000100 and decompresses data accordingly (without
taking the size of allocated buffer), it is possible to overflow the allocated heap
during the decompression process.

It is probably challenging to exploit this in code compiled into firmware as you
need to control input in the first place, but the LzmaUefiDecompressGetInfo
function may be used to 3rd party software that parses UEFI firmware image files.
For example, UEFITool has its own copy of this function (and fixed it by themselves
https://github.com/LongSoft/UEFITool/commit/f507d71ead4dcc516d6be1b5b3cd61f900abb4a4#diff-
fa8944f45e026a9e683e0f552b418626).

I do not have an PoC firmware image and a standalone parser to demonstrate the
issue handy, but hopefully, the issue is clear.
```

Vincent Zimmer 2019-05-29 12:46:37 UTC Comment 1

```
5/29/19 tianocore infosec bugscrub - brent h. to dig in further.
Kinney - looks like > 4gb input data would cause issue
Brett - ok if 32-bit everywhere.  mixing would be an issue
```

Bret Barkelew 2019-05-29 12:47:14 UTC Comment 2

```
Bret*
;)
```

brent.holtsclaw 2019-05-29 17:41:05 UTC Comment 3

```
It appears to be a security issue if using compilers size_t on a 64bit system.
Suggest either casting to SizeT within LzmaUefiDecompressGetInfo or casting to
UINT32 within LzmaUefiDecompress.

LzmaUefiDecompressGetInfo function casts DestinationSize to UINT32
..
DecodedSize = GetDecodedSizeOfBuf((UINT8*)Source);
*DestinationSize = (UINT32)DecodedSize;

LzmaUefiDecompress function casts DecodedBufSize to SizeT which appears to also be
defined as UINTN
DecodedBufSize = (SizeT)GetDecodedSizeOfBuf((UINT8*)Source);
It then uses the DecodedBufSize in call to LzmaDecode
  LzmaResult = LzmaDecode(
    Destination,
    &DecodedBufSize,
    (Byte*)((UINT8*)Source + LZMA_HEADER_SIZE),
    &EncodedDataSize,
    Source,
    LZMA_PROPS_SIZE,
    LZMA_FINISH_END,
    &Status,
    &(AllocFuncs.Functions)
    )

Function LzmaDecode uses DecodedBufSize as the size of the destination buffer and
also treats it as SizeT
SRes LzmaDecode(Byte *dest, SizeT *destLen, const Byte *src, SizeT *srcLen,
    const Byte *propData, unsigned propSize, ELzmaFinishMode finishMode,
    ELzmaStatus *status, ISzAlloc *alloc)

SizeT outSize = *destLen
p.dicBufSize = outSize;
res = LzmaDec_DecodeToDic(&p, outSize, src, srcLen, finishMode, status);
```

Bret Barkelew 2019-06-25 16:48:12 UTC Comment 4

```
Agree with Brent about explicitly casting consistently in this library.
```

Laszlo Ersek 2019-08-01 19:19:02 UTC Comment 5

(comment made against edk2 @ 3d34b5f32692)

I think I disagree about both impact (comment 0) and mitigation
(comments 3-4).

Regarding impact: 3rd party UEFI binaries may be LZMA compressed. They
are decompressed before e.g. secure boot image verification. This means
that a crafted EFI binary can look small but cause buffer overflow in
section extraction, before validation. A similar issue was ~~bug 686~~,
which merited multiple CVE numbers.

(In particular, see:
- ~~https://bugzilla.tianocore.org/show_bug.cgi?id=686#c13~~
- https://lists.01.org/pipermail/edk2-devel/2018-October/031095.html
- https://lists.01.org/pipermail/edk2-devel/2018-October/031138.html
)

Regarding mitigation:

- I don't think we should ever truncate to UINT32 if he input is
  provided as UINT64. (As a special case, I don't think we should
  truncate to UINT32 within LzmaUefiDecompress().)

- Outputting UINT64 from LzmaUefiDecompressGetInfo() is theoretically
  safe, but messy in practice. While LzmaUefiDecompressGetInfo() is
  indeed an internal function, declared in:

    MdeModulePkg/Library/LzmaCustomDecompressLib/LzmaDecompressLibInternal.h

  its output parameter called "DestinationSize" is directly propagated
  out of two public functions. Namely (all pathnames relative to
  "MdeModulePkg/Library/LzmaCustomDecompressLib"):

  - LzmaGuidedSectionGetInfo()   [GuidedSectionExtraction.c]
  - LzmaArchGuidedSectionGetInfo [F86GuidedSectionExtraction.c]

  Both of these functions are registered with the
  ExtractGuidedSectionRegisterHandlers() API of the
  ExtractGuidedSectionLib class.

  The registered functions need to conform to the
  EXTRACT_GUIDED_SECTION_GET_INFO_HANDLER type.

  The registered functions are searched for, and called, through the
  public ExtractGuidedSectionGetInfo() API, from:

    MdePkg/Include/Library/ExtractGuidedSectionLib.h

  The ExtractGuidedSectionGetInfo() function is called from several
  places in edk2. I guess the most important could be that the DXE Core
  uses ExtractGuidedSectionGetInfo() for implementing
  EFI_GUIDED_SECTION_EXTRACTION_PROTOCOL -- see
  "MdeModulePkg/Core/Dxe/SectionExtraction/CoreSectionExtraction.c".

  This protocol comes from the PI spec, and the "OutputSize" output
  parameter has type UINTN. Therefore, even if all the internal
  propagation of the original "DecodedSize" were widened correctly to
  64-bit, the PI protocol would still be a bottleneck on 32-bit
  architectures.

  Therefore, I would catch >= 4GB DestinationSize values right in
  LzmaUefiDecompressGetInfo(), and return an error. The function's
  return type is already RETURN_STATUS, and all of the call sites
  correctly propagate errors. This wouldn't enable edk2 to extract
  GUIDed (LZMA) sections larger than 4GB, but we'd replace the potential
  buffer overflow with clean errors, at minimal cost to dependent code.
  (4GB+ GUIDed sections are clearly not needed in practice.)

Laszlo Ersek 2019-08-09 09:42:31 UTC Comment 6

The CustomGuidedSectionExtract() function in
"MdeModulePkg/Core/Dxe/SectionExtraction/CoreSectionExtraction.c" at commit
4b1b7c191309 is vulnerable to the exact issue described in comment 0; therefore I'm
marking this as CONFIRMED.

```
  UINT32          OutputBufferSize;
...
  Status = ExtractGuidedSectionGetInfo (
             InputSection,
             &OutputBufferSize,
             &ScratchBufferSize,
             &SectionAttribute
             );
...
  if (OutputBufferSize > 0) {
    //
    // Allocate output buffer
    //
    AllocatedOutputBuffer = AllocatePool (OutputBufferSize);
...
    *OutputBuffer = AllocatedOutputBuffer;
  }
...
  Status = ExtractGuidedSectionDecode (
             InputSection,
             OutputBuffer,
             ScratchBuffer,
             AuthenticationStatus
             );
```

and see comment 3 for where the last function call quoted above ends.

Alex Ionescu 2020-07-01 16:29:05 UTC Comment 7

Any plan on actually fixing this issue? It seems to be bouncing around. Since it
seems to have real security merit, perhaps we should plan a disclosure timeline
since many 3rd party libraries are affected.

Laszlo Ersek 2020-09-25 11:30:39 UTC Comment 8

I'll try to propose a patch (as an attachment) next week.

Laszlo Ersek 2020-09-28 12:22:25 UTC Comment 9

Created attachment 564 [details]
[PATCH] MdeModulePkg/LzmaCustomDecompressLib: catch 4GB+ uncompressed buffer sizes

Proposed patch, applies on top of commit 1d058c3e86b0 ("IntelFsp2Pkg GenCfgOpt.py:
Initialize IncLines as empty list", 2020-09-25).

Laszlo Ersek 2020-09-28 12:26:06 UTC Comment 10

(In reply to Laszlo Ersek from comment #9)
> Created attachment 564 [details]
> [PATCH] MdeModulePkg/LzmaCustomDecompressLib: catch 4GB+ uncompressed buffer
> sizes
>
> Proposed patch, applies on top of commit 1d058c3e86b0 ("IntelFsp2Pkg
> GenCfgOpt.py: Initialize IncLines as empty list", 2020-09-25).

Cc: Dandan Bi <dandan.bi@intel.com>

Cc: Hao A Wu <hao.a.wu@intel.com>
Cc: Jian J Wang <jian.j.wang@intel.com>
Cc: Liming Gao <gaoliming@byosoft.com.cn>
Cc: Philippe Mathieu-Daudé <philmd@redhat.com>

Please review.

Note: I have only regression-tested this patch, using OVMF's automatic LZMA
decompression (covering PEIFV and DXEFV) during boot, and S3 resume. (See the
message on commit b24fca05751f for a few more details on PEIFV and DXEFV.)

Alex Ionescu 2020-09-28 14:46:34 UTC Comment 11

Thanks for the patch.

Team/Vincent -- would it help to get a CVE assigned? I can work with MITRE on
that...

gaoliming 2020-09-28 21:29:37 UTC Comment 12

I agree this fix. Reviewed-by: Liming Gao <gaoliming@byosoft.com.cn>

Laszlo Ersek 2020-09-29 14:56:01 UTC Comment 13

Hi Alex,

(In reply to Alex Ionescu from comment #11)
> Thanks for the patch.
>
> Team/Vincent -- would it help to get a CVE assigned? I can work with MITRE
> on that...

I agree that a CVE should be assigned (see my reference to ~~bug#686~~ in comment#5
above).

Luckily, TianoCore has recently become a CNA
<https://cve.mitre.org/cve/request_id.html#t> thanks to the work of the infosec
group. From Vincent's announcement on the edk2-infosec mailing list, "Points of
contact for infosec and CVE's will include Kevin Davis from Insyde, Eric Johnson
from AMI, Dick Wilkins from Phoenix, and Vincent Zimmer from Intel". So (I think?)
we need not contact an external org for assigning a CVE.

Thanks.

Laszlo Ersek 2020-09-29 14:58:27 UTC Comment 14

(In reply to gaoliming from comment #12)
> I agree this fix. Reviewed-by: Liming Gao <gaoliming@byosoft.com.cn>

Thank you, Liming!

So if there are no more comments on the patch, I think we should set up an embargo
end date, so that people can start applying the patch.

Once the embargo elapses, I'll post the patch to edk2-devel for public review, with
your R-b from comment#12 included. Thanks!

Alex Ionescu 2020-09-29 17:23:31 UTC Comment 15

(In reply to Laszlo Ersek from comment #13)
> Hi Alex,
>
> (In reply to Alex Ionescu from comment #11)
> > Thanks for the patch.
> >
> > Team/Vincent -- would it help to get a CVE assigned? I can work with MITRE
> > on that...
>
> I agree that a CVE should be assigned (see my reference to ~~bug#686~~ in
> comment#5 above).
>
> Luckily, TianoCore has recently become a CNA
> <https://cve.mitre.org/cve/request_id.html#t> thanks to the work of the
> infosec group. From Vincent's announcement on the edk2-infosec mailing list,
> "Points of contact for infosec and CVE's will include Kevin Davis from
> Insyde, Eric Johnson from AMI, Dick Wilkins from Phoenix, and Vincent Zimmer
> from Intel". So (I think?) we need not contact an external org for assigning
> a CVE.
>
> Thanks.

That's awesome. Let me know if this is something I need to ping Vincent on or if it
will be handled automatically as part of the bug/triage process. I want to make
sure the bug will be correctly credited to Satoshi Tanda -- CrowdStrike and to be
aware of disclosure timeline so that we can also absorb the patch in our own
product -- for now we used a workaround to avoid showcasing the issue (it's a
closed-source component).

Laszlo Ersek 2020-10-01 03:33:42 UTC Comment 16

According to the flowchart at

https://github.com/tianocore/tianocore.github.io/wiki/Reporting-Security-Issues

(direct link:
<https://raw.githubusercontent.com/jwang36/tianocore.github.io/master/security/flowchart.svg>)

from phase 2, we're still missing the CVSS evaluation step on this BZ. Once that is
done (or may in parallel to it), we should indeed figure out the embargo length
(phase 3), and get a CVE number (for phase 4).

I'm going to send an email to the edk2-infosec mailing list now, CC'ing the CVE
contacts, about this BZ. Thanks.

Philippe Mathieu-Daudé 2020-10-07 07:17:09 UTC Comment 17

(In reply to Laszlo Ersek from comment #9)
> Created attachment 564 [details]
> [PATCH] MdeModulePkg/LzmaCustomDecompressLib: catch 4GB+ uncompressed buffer
> sizes
>
> Proposed patch, applies on top of commit 1d058c3e86b0 ("IntelFsp2Pkg
> GenCfgOpt.py: Initialize IncLines as empty list", 2020-09-25).

Reviewed-by: Philippe Mathieu-Daude <philmd@redhat.com>

Laszlo Ersek 2020-10-07 11:52:56 UTC Comment 18

Thank you, Phil!

Laszlo Ersek 2020-11-12 12:16:52 UTC Comment 19

PROPOSED PUBLIC DATE (opening up the BZ and posting the patch to edk2-devel):

  Thursday 2020-Nov-19 07:00 UTC

in order to get the fix into edk2-stable202011.

Riccardo Schirone 2020-11-13 06:14:53 UTC Comment 20

Is this going to have a CVE before it goes public?

Laszlo Ersek 2020-11-13 15:21:13 UTC Comment 21

(In reply to Riccardo Schirone from comment #20)
> Is this going to have a CVE before it goes public?

CC'ing Eric.

Laszlo Ersek 2020-11-19 06:53:01 UTC Comment 22

(In reply to Laszlo Ersek from comment #9)
> Created attachment 564 [details]
> [PATCH] MdeModulePkg/LzmaCustomDecompressLib: catch 4GB+ uncompressed buffer
> sizes
>
> Proposed patch, applies on top of commit 1d058c3e86b0 ("IntelFsp2Pkg
> GenCfgOpt.py: Initialize IncLines as empty list", 2020-09-25).

Public posting:

* [edk2-devel] [PATCH RESEND 0/1]
  security fix: possible heap corruption with LzmaUefiDecompressGetInfo

msgid <20201119115034.12897-1-lersek@redhat.com>
https://edk2.groups.io/g/devel/message/67708
https://www.redhat.com/archives/edk2-devel-archive/2020-November/msg00866.html

Alex Ionescu 2020-11-19 11:25:12 UTC Comment 23

@Ericj -- what's the CVE/Ack for this? I see this is now public.

Laszlo Ersek 2020-11-20 21:06:10 UTC Comment 24

Merged as commit e7bd0dd26db7, via <https://github.com/tianocore/edk2/pull/1138>.

kevinj 2021-03-03 11:46:54 UTC Comment 25

Created attachment 658 [details]
CVE .json file

I have attached the .json file for CVE classification. Please review and provide feedback, especially for the version.

Laszlo Ersek 2021-03-03 12:02:57 UTC Comment 26

Hello Kevin,

last vulnerable release: edk2-stable202008
first fixed release: edk2-stable202011

$ git tag --contains e7bd0dd26db7
edk2-stable202011

Laszlo Ersek 2021-03-03 12:05:23 UTC Comment 27

Upon reviewing commit e7bd0dd26db7, I think CWE-122 is a good match.

kevinj 2021-03-04 14:29:59 UTC Comment 28

Created attachment 666 [details]
CVE .json file_2

Thank you Laszlo for your feedback. I have updated the version in the .json file and re-uploaded it.

Alex Ionescu 2021-03-10 09:26:36 UTC Comment 29

Hi,

Still wondering when it's possible to have the CVE assigned? This has almost taken two years now since reporting.

kevinj 2021-03-11 16:59:03 UTC Comment 30

(In reply to Alex Ionescu from comment #29)
> Hi,
>
> Still wondering when it's possible to have the CVE assigned? This has almost
> taken two years now since reporting.

We have requested CVE-IDs from MITRE. We are still waiting to receive said ID number from them.

Laszlo Ersek 2021-03-12 16:03:49 UTC Comment 31

Thanks for the CVE number, Kevin!

kevinj 2021-03-12 16:05:15 UTC Comment 32

Laszlo,

Please review the .json file again, especially the version this bug is observed in and inform me when you plan to publicly disclose this bug, so we know when to submit this CVE back to MITRE. Thank you!

Laszlo Ersek 2021-03-12 16:56:54 UTC Comment 33

The release info (edk2-stable202008) in the JSON from comment 28 seems OK, matching my comment 26.

Regarding public disclosure, we're past that -- please see comment 19 and comment 22. The issue was disclosed on November 19, 2020.

Alex Ionescu 2021-03-12 17:09:26 UTC Comment 34

Since TianoCore is a CNA, I'm unclear why you're contacting MITRE to get a CVE -- can't Tiano issue one directly?

Additionally, when will the information be published here:

https://edk2-docs.gitbook.io/security-advisory/

Satoshi Tanda 2021-04-18 14:27:24 UTC Comment 35

Just noticed CVE-2021-28211 was assigned. It is still reserved in MITRE, but some vendors have public info
https://access.redhat.com/security/cve/CVE-2021-28211

Kim Olsun 2021-09-22 15:47:43 UTC Comment 36

http://www.compilatori.com/tech/nvidia-and-samsung/
http://www.wearelondonmade.com/tech/nvidia-and-samsung/
http://www.lopspeech.com/tech/nvidia-and-samsung/
http://joerg.li/tech/nvidia-and-samsung/ http://connstr.net/tech/nvidia-and-samsung/ http://embermanchester.uk/tech/nvidia-and-samsung/
http://www.slipstone.co.uk/tech/nvidia-and-samsung/
http://www.logoarts.co.uk/tech/nvidia-and-samsung/
http://www.acpirateradio.co.uk/health/covid-and-tech/
https://waytowhatsnext.com/category/services/ https://www.webb-dev.co.uk/category/shopping/ http://www.lu-bloomington.com/category/services/
http://www-look-4.com/tech/nvidia-and-samsung/