Bug 2046194 - CVE-2022-0485 libnbd: nbdcopy ignore read and write errors - destination image corrupted [rhel-9.0]

Keywords: Security × SecurityTracking × Triaged ×

Status: CLOSED ERRATA

Alias: None

Product: Red Hat Enterprise Linux 9

Component: libnbd

Version: CentOS Stream

Hardware: Unspecified

OS: Unspecified

Priority: high

Severity: medium

Target Milestone: rc

Target Release: ---

Assignee: Richard W.M. Jones

QA Contact: Vera

Docs Contact:

URL:

Whiteboard:

Depends On:

Blocks: 2045718 ~~CVE-2022-0485~~

TreeView+ depends on / blocked

Reported: 2022-01-26 11:28 UTC by Nir Soffer

Modified: 2022-05-17 13:13 UTC (History)

CC List: 12 users (show)

Fixed In Version: libnbd-1.10.5-1.el9

Doc Type: If docs needed, set a value

Doc Text:

Clone Of:

Environment:

Last Closed: 2022-05-17 12:51:02 UTC

Type: Bug

Target Upstream Version:

Dependent Products:

Attachments (Terms of Use)
Add an attachment (proposed patch, testcase, etc.)

Links

System	ID	Private	Priority	Status	Summary	Last Updated
Red Hat Issue Tracker	RHELPLAN-109801	0	None	None	None	2022-01-26 11:29:56 UTC
Red Hat	RHEA-2022:2409	0	None	None	None	2022-05-17

Product Errata						12:51:16 UTC

Nir Soffer 2022-01-26 11:28:56 UTC Description

```
Description of problem:

When copying from NBD server using the asynchronous copy mode
(default)
nbdcopy may create a corrupted destination image if read or
write NBD
command start but the server returns an error. nbdcopy also
exit with
zero exit code, so programs running it cannot detect that the
operation
failed.

Version-Release number of selected component (if applicable):
libnbd-1.10.3-1.el9.x86_64

How reproducible:
Always

Steps to Reproduce:

Reproducing read errors:

1. Create source image

$ dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "B" >
src.img

$ hexdump -C src.img
00000000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000

2. Create destination image

$ dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "A" >
dst.img

$ hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00400000

3. Start nbdkit for reading from source image

$ nbdkit -f -v python error.py file=src.img
nbdkit: debug: nbdkit 1.28.4 (nbdkit-1.28.4-2.el9)
nbdkit: debug: TLS disabled: could not load TLS certificates
nbdkit: debug: registering plugin
/usr/lib64/nbdkit/plugins/nbdkit-python-plugin.so
nbdkit: debug: registered plugin
/usr/lib64/nbdkit/plugins/nbdkit-python-plugin.so (name
python)
nbdkit: debug: python: load
nbdkit: debug: python: config key=script, value=error.py
nbdkit: debug: module requested API_VERSION 2
nbdkit: debug: python: config key=file, value=src.img
```

```
nbdkit: debug: python: config_complete
nbdkit: debug: using thread model: serialize_all_requests
nbdkit: debug: python: get_ready
nbdkit: debug: bound to IP address <any>:10809 (2 socket(s))
nbdkit: debug: python: after_fork

4. Copy from nbdkit to destination image

$ nbdcopy nbd://localhost dst.img; echo $?
0

5. Check nbdkit log

nbdkit: debug: accepted connection
nbdkit: python[1]: debug: python: preconnect
nbdkit: python[1]: debug: newstyle negotiation: flags: global
0x3
nbdkit: python[1]: debug: newstyle negotiation: client flags:
0x3
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_STRUCTURED_REPLY: client requested structured replies
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_SET_META_CONTEXT: client requested export ''
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_SET_META_CONTEXT: set count: 1
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_SET_META_CONTEXT: set base:allocation
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_SET_META_CONTEXT: replying with base:allocation id 1
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_SET_META_CONTEXT: reply complete
nbdkit: python[1]: debug: newstyle negotiation: NBD_OPT_GO:
client requested export ''
nbdkit: python[1]: debug: python: open readonly=0
exportname="" tls=0
nbdkit: python[1]: debug: python: default_export readonly=0
tls=0
nbdkit: python[1]: debug: python: open returned handle
0x7f12740016b0
nbdkit: python[1]: debug: python: prepare readonly=0
nbdkit: python[1]: debug: python: get_size
nbdkit: python[1]: debug: python: can_write
nbdkit: python[1]: debug: python: can_zero
nbdkit: python[1]: debug: python: can_fast_zero
nbdkit: python[1]: debug: python: can_trim
nbdkit: python[1]: debug: python: can_fua
nbdkit: python[1]: debug: python: can_flush
nbdkit: python[1]: debug: python: is_rotational
nbdkit: python[1]: debug: python: can_multi_conn
nbdkit: python[1]: debug: python: can_cache
nbdkit: python[1]: debug: python: can_extents
nbdkit: python[1]: debug: newstyle negotiation: flags: export
0x8c1
nbdkit: python[1]: debug: newstyle negotiation: NBD_OPT_GO:
ignoring NBD_INFO_* request 3 (NBD_INFO_BLOCK_SIZE)
nbdkit: python[1]: debug: handshake complete, processing
requests serially
nbdkit: python[1]: debug: python: extents count=4194304
offset=0 req_one=0
nbdkit: python[1]: debug: python: pread count=262144 offset=0
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
```

```
 RuntimeError: pread error offset=0 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=262144
nbdkit: python[1]: debug: python: pread count=262144
offset=524288
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=524288 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=786432
nbdkit: python[1]: debug: python: pread count=262144
offset=1048576
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=1048576 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=1310720
nbdkit: python[1]: debug: python: pread count=262144
offset=1572864
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=1572864 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=1835008
nbdkit: python[1]: debug: python: pread count=262144
offset=2097152
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=2097152 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=2359296
nbdkit: python[1]: debug: python: pread count=262144
offset=2621440
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
```

```
 RuntimeError: pread error offset=2621440 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=2883584
nbdkit: python[1]: debug: python: pread count=262144
offset=3145728
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=3145728 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=3407872
nbdkit: python[1]: debug: python: pread count=262144
offset=3670016
nbdkit: python[1]: error: error.py: pread: error: Traceback
(most recent call last):
   File "error.py", line 33, in pread
    raise RuntimeError(f"pread error offset={offset} count=
{len(buf)}")
 RuntimeError: pread error offset=3670016 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pread count=262144
offset=3932160
nbdkit: python[1]: debug: client sent NBD_CMD_DISC, closing
connection
nbdkit: python[1]: debug: python: finalize
nbdkit: python[1]: debug: python: close
```

6. Check destination image

```
$ hexdump -C dst.img
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
00040000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00080000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
000c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00100000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
00140000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00180000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|................|
*
001c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00200000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

```
|.................|
*
00240000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00280000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|.................|
*
002c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00300000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|.................|
*
00340000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00380000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
|.................|
*
003c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000
```

nbdcopy ignored the read errors. Failed reads are written as zeroes.

Reproducing write errors:

1. Create destination image

```
$ dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "A" >
dst.img

$ hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00400000
```

2. Create source image

```
$ dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "B" >
src.img

$ hexdump -C src.img
00000000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000
```

3. Run nbdkit for writing into destination image

```
$ nbdkit -f -v python error.py file=dst.img
nbdkit: debug: nbdkit 1.28.4 (nbdkit-1.28.4-2.el9)
nbdkit: debug: TLS disabled: could not load TLS certificates
nbdkit: debug: registering plugin
/usr/lib64/nbdkit/plugins/nbdkit-python-plugin.so
nbdkit: debug: registered plugin
/usr/lib64/nbdkit/plugins/nbdkit-python-plugin.so (name
python)
nbdkit: debug: python: load
```

```
nbdkit: debug: python: config key=script, value=error.py
nbdkit: debug: module requested API_VERSION 2
nbdkit: debug: python: config key=file, value=dst.img
nbdkit: debug: python: config_complete
nbdkit: debug: using thread model: serialize_all_requests
nbdkit: debug: python: get_ready
nbdkit: debug: bound to IP address <any>:10809 (2 socket(s))
nbdkit: debug: python: after_fork

4. Copy source image to nbdkit

$ nbdcopy src.img nbd://localhost; echo $?
0

5. Check nbdkit log

nbdkit: debug: accepted connection
nbdkit: python[1]: debug: python: preconnect
nbdkit: python[1]: debug: newstyle negotiation: flags: global
0x3
nbdkit: python[1]: debug: newstyle negotiation: client flags:
0x3
nbdkit: python[1]: debug: newstyle negotiation:
NBD_OPT_STRUCTURED_REPLY: client requested structured replies
nbdkit: python[1]: debug: newstyle negotiation: NBD_OPT_GO:
client requested export ''
nbdkit: python[1]: debug: python: open readonly=0
exportname="" tls=0
nbdkit: python[1]: debug: python: default_export readonly=0
tls=0
nbdkit: python[1]: debug: python: open returned handle
0x7ff134001460
nbdkit: python[1]: debug: python: prepare readonly=0
nbdkit: python[1]: debug: python: get_size
nbdkit: python[1]: debug: python: can_write
nbdkit: python[1]: debug: python: can_zero
nbdkit: python[1]: debug: python: can_fast_zero
nbdkit: python[1]: debug: python: can_trim
nbdkit: python[1]: debug: python: can_fua
nbdkit: python[1]: debug: python: can_flush
nbdkit: python[1]: debug: python: is_rotational
nbdkit: python[1]: debug: python: can_multi_conn
nbdkit: python[1]: debug: python: can_cache
nbdkit: python[1]: debug: python: can_extents
nbdkit: python[1]: debug: newstyle negotiation: flags: export
0x8c1
nbdkit: python[1]: debug: newstyle negotiation: NBD_OPT_GO:
ignoring NBD_INFO_* request 3 (NBD_INFO_BLOCK_SIZE)
nbdkit: python[1]: debug: handshake complete, processing
requests serially
nbdkit: python[1]: debug: python: pwrite count=262144 offset=0
fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=0 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=262144 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=524288 fua=0
```

```
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=524288 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=786432 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=1048576 fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=1048576 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=1310720 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=1572864 fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=1572864 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=1835008 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=2097152 fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=2097152 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=2359296 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=2621440 fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=2621440 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=2883584 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=3145728 fua=0
```

```
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=3145728 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=3407872 fua=0
nbdkit: python[1]: debug: python: pwrite count=262144
offset=3670016 fua=0
nbdkit: python[1]: error: error.py: pwrite: error: Traceback
(most recent call last):
   File "error.py", line 43, in pwrite
    raise RuntimeError(f"pwrite error offset={offset} count=
{len(buf)}")
 RuntimeError: pwrite error offset=3670016 count=262144

nbdkit: python[1]: debug: sending error reply: Input/output
error
nbdkit: python[1]: debug: python: pwrite count=262144
offset=3932160 fua=0
nbdkit: python[1]: debug: client sent NBD_CMD_DISC, closing
connection
nbdkit: python[1]: debug: python: finalize
nbdkit: python[1]: debug: python: close
```

6. Check destination image

```
$ hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00040000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00080000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
000c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00100000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00140000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00180000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
001c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00200000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00240000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00280000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
```

```
*
002c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00300000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00340000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00380000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
003c0000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000
```

nbdcopy ignored the write errors. The destination image contains mix of new
and old data.

Actual results:

In both case nbdcopy exit with zero exit code creating corrupted image.

Expected results:

In both cases nbdcopy should fail with non-zero exit code on the first error.

Nir Soffer 2022-01-26 11:29:55 UTC Comment 1

Created attachment 1855494 [details]
nbdkit plugin for injecting read and write errors

Eric Blake 2022-02-03 01:57:56 UTC Comment 2

Upstream patch proposed:
https://listman.redhat.com/archives/libguestfs/2022-February/msg00039.html

Richard W.M. Jones 2022-02-05 10:40:57 UTC Comment 3

Hello Vera, would it possible for you to QA ACK this please when you are back from holiday.

Richard W.M. Jones 2022-02-05 10:42:09 UTC Comment 4

https://gitlab.com/redhat/centos-stream/rpms/libnbd/-/merge_requests/14
(Waiting on QA ACK)

Vera 2022-02-08 03:47:58 UTC Comment 5

Verified with libnbd-1.10.4-1.el9.x86_64

Reproduced with libnbd-1.10.3-1.el9.x86_64
Steps:
1. Create source image
```
# dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "B" >
src.img
# hexdump -C src.img
00000000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000
```

2. Create destination image
```
# dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "A" >
dst.img
# hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00400000
```

3. Copy from nbdkit to destination image
```
# nbdcopy nbd://localhost dst.img; echo $?
0
```

4. Check destination image
```
# hexdump -C dst.img
......
002655c0  65 5f 65 78 74 34 5f 5f  6d 62 61 6c 6c 6f 63 0a
|e_ext4__mballoc.|
002655d0  66 66 66 66 66 66 66 66  38 31 33 63 32 36 35 30
|ffffffff813c2650|
002655e0  20 74 20 5f 5f 62 70 66  5f 74 72 61 63 65 5f 65  |
t __bpf_trace_e|
002655f0  78 74 34 5f 64 69 72 65  63 74 5f 49 4f 5f 65 78
|xt4_direct_IO_ex|
00265600  69 74 0a 66 66 66 66 66  66 66 66 38 31 33 63 32
|it.ffffffff813c2|
00265610  36 36 30 20 74 20 5f 5f  62 70 66 5f 74 72 61 63
|660 t __bpf_trac|
00265620  65 5f 65 78 74 34 5f 65  78 74 5f 68 61 6e 64 6c
|e_ext4_ext_handl|
00265630  65 5f 75 6e 77 72 69 74  74 65 6e 5f 65 78 74 65
|e_unwritten_exte|
00265640  6e 74 73 0a 66 66 66 66  66 66 66 66 38 31 33 63
|nts.ffffffff813c|
00265650  32 36 37 30 20 74 20 5f  5f 62 70 66 5f 74 72 61
|2670 t __bpf_tra|
00265660  63 65 5f 65 78 74 34 5f  72 65 6d 6f 76 65 5f 62
|ce_ext4_remove_b|
00265670  6c 6f 63 6b 73 0a 66 66  66 66 66 66 66 66 38 31
|locks.ffffffff81|
00265680  33 63 32 36 38 30 20 74  20 5f 5f 62 70 66 5f 74
|3c2680 t __bpf_t|
```

```
00265690  72 61 63 65 5f 65 78 74  34 5f 65 73 5f 73 68 72
|race_ext4_es_shr|
002656a0  69 6e 6b 0a 66 66 66 66  66 66 66 66 38 31 33 63
|ink.ffffffff813c|
002656b0  32 36 39 30 20 74 20 5f  5f 62 70 66 5f 74 72 61
|2690 t __bpf_tra|
002656c0  63 65 5f 65 78 74 34 5f  66 69 6e 64 5f 64 65 6c
|ce_ext4_find_del|
002656d0  61 6c 6c 6f 63 5f 72 61  6e 67 65 0a 66 66 66 66
|alloc_range.ffff|
......
```

Verified with libnbd-1.10.4-1.el9.x86_64

Steps:
1. Create source image
```
# dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "A" >
dst.img
# hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00400000
```

2. Create destination image
```
# dd if=/dev/zero bs=1M count=4 status=none | tr "\0" "B" >
src.img
# hexdump -C src.img
00000000  42 42 42 42 42 42 42 42  42 42 42 42 42 42 42 42
|BBBBBBBBBBBBBBBB|
*
00400000
```

3. Copy from nbdkit to destination image
```
# nbdcopy nbd://localhost dst.img; echo $?
0
```

4. Check destination image
```
# hexdump -C dst.img
00000000  41 41 41 41 41 41 41 41  41 41 41 41 41 41 41 41
|AAAAAAAAAAAAAAAA|
*
00400000
```

Moving to verified:Tested

Richard W.M. Jones 2022-02-08 08:40:31 UTC Comment 8

Hi Vera, I don't understand the reproducer. Is there a missing
step between 2. and 3.? ie. a step where you run an NBD server?

Anyway I have a much simpler, one-line reproducer:

```
$ nbdkit -U - eval get_size=' echo 100M ' pread=' if [ $4 -lt
10000000 ]; then dd if=/dev/zero count=$3 iflag=count_bytes ;
else echo EIO Failed >&2 ; exit 1; fi ' --run 'nbdcopy $uri
null: -p'
```

For libnbd-1.10.3-1.el9.x86_64:

```
...
nbdkit: eval[1]: error: /tmp/nbdkitDT21TP/pread: Failed
nbdkit: eval[1]: error: /tmp/nbdkitDT21TP/pread: Failed
nbdkit: eval[1]: error: /tmp/nbdkitDT21TP/pread: Failed
nbdkit: eval[1]: error: /tmp/nbdkitDT21TP/pread: Failed
 100% [************************************]
$ echo $?
0
```

Notice that even though the server sent errors to nbdcopy, it succeeded.

For libnbd-1.10.4-1.el9.x86_64:

```
$ nbdkit -U - eval get_size=' echo 100M ' pread=' if [ $4 -lt
10000000 ]; then dd if=/dev/zero count=$3 iflag=count_bytes ;
else echo EIO Failed >&2 ; exit 1; fi ' --run 'nbdcopy $uri
null: -p'
nbdkit: eval[1]: error: /tmp/nbdkitQtKlRw/pread: Failed
read at offset 10223616 failed: Input/output error
nbdkit: eval[1]: error: /tmp/nbdkitQtKlRw/pread: Failed
nbdkit: eval[1]: error: write error reply: Broken pipe
$ echo $?
1
```

The command correctly fails with the new nbdcopy.

Vera 2022-02-09 10:28:30 UTC Comment 9

Verified with the version: libnbd-1.10.4-1.el9.x86_64

```
# nbdkit -U - eval get_size=' echo 100M ' pread=' if [ $4 -lt
10000000 ]; then dd if=/dev/zero count=$3 iflag=count_bytes ;
else echo EIO Failed >&2 ; exit 1; fi ' --run 'nbdcopy $uri
null: -p'
nbdkit: eval[1]: error: /tmp/nbdkit0BrU0q/pread: Failed
read at offset 10223616 failed: Input/output error
nbdkit: eval[1]: error: /tmp/nbdkit0BrU0q/pread: Failed
nbdkit: eval[1]: error: /tmp/nbdkit0BrU0q/pread: Failed
nbdkit: eval[1]: error: write error reply: Broken pipe
# echo $?
1
```

Moving to Verified.

Richard W.M. Jones 2022-02-10 16:16:44 UTC Comment 10

Vera, FYI libnbd-1.10.5-1.el9 contains some extra fixes which are
related to this CVE:

https://gitlab.com/nbdkit/libnbd/-/commit/e15864c364aef710a7826b2b25c88a360f9819d6
https://gitlab.com/nbdkit/libnbd/-/commit/56d2611bd6fcdb559ee5ff11532dec75eb2f8472

https://gitlab.com/nbdkit/libnbd/-/commit/c97b12493c01b09b4faf41120cc11cf03780b51b
https://gitlab.com/nbdkit/libnbd/-/commit/c79706af4e7475bf58861a143b77b77a54e7a1cd

If you want to test these, maybe the only thing to test is that the two new
APIs appear (nbd_set_pread_initialize, nbd_get_pread_initialize), and the
libnbd-security(3) man page has been updated with the CVE.

Vera 2022-02-11 13:17:21 UTC Comment 11

Verified with the version: libnbd-1.10.5-1.el9.x86_64

1. Verify the nbdcopy:

```
# nbdkit -U - eval get_size=' echo 100M ' pread=' if [ $4 -lt
10000000 ]; then dd if=/dev/zero count=$3 iflag=count_bytes ;
else echo EIO Failed >&2 ; exit 1; fi ' --run 'nbdcopy $uri
null: -p'
nbdkit: eval[1]: error: /tmp/nbdkitsjMPHS/pread: Failed
read at offset 10223616 failed: Input/output error
nbdkit: eval[1]: error: /tmp/nbdkitsjMPHS/pread: Failed
nbdkit: eval[1]: error: /tmp/nbdkitsjMPHS/pread: Failed
nbdkit: eval[1]: error: write error reply: Broken pipe
# echo $?
1
```

2. Verify the two new APIs:(nbd_aio_command_completed, nbd_get_pread_initialize) appear in the nbdsh;

```
# nbdsh

Welcome to nbdsh, the shell for interacting with
Network Block Device (NBD) servers.

The ‘nbd’ module has already been imported and there
is an open NBD handle called ‘h’.

h.connect_tcp("remote", "10809")   # Connect to a remote
server.
h.get_size()                       # Get size of the remote
disk.
buf = h.pread(512, 0, 0)           # Read the first sector.
exit() or Ctrl-D                   # Quit the shell
help(nbd)                          # Display documentation

nbd> help(nbd)
....
     |  aio_command_completed(self, cookie)
     |      ▶ check if the command completed
     |
     |      Return true if the command completed. If this
function
     |      returns true then the command was successful and
it has
     |      been retired. Return false if the command is still
in
     |      flight. This can also fail with an error in case
the
```

```
     |      command failed (in this case the command is also
     |      retired). A command is retired either via this
command,
     |      or by using a completion callback which returns 1.
     |
     |      The "cookie" parameter is the positive unique 64
bit
     |      cookie for the command, as returned by a call such
as
     |      "nbd.aio_pread".
......

     |  get_pread_initialize(self)
     |      ▶ see whether libnbd pre-initializes read buffers
     |
     |      Return whether libnbd performs a pre-
initialization of a
     |      buffer passed to "nbd.pread" and similar to all
zeroes,
     |      as set by "nbd.set_pread_initialize".
....


3. Check the doc on the CVE;

Download the src pkg from brew:
# ls
libnbd-1.10.5-1.el9.src.rpm

check the docs:
# rpmbuild -rp libnbd-1.10.5-1.el9.src.rpm

# cat /root/rpmbuild/BUILD/libnbd-1.10.5/docs/libnbd-
security.pod |grep 2022 -A 5
=head2 CVE-2022-0485
silent data corruption when using L<nbdcopy(1)>

See the full announcement here:
L<https://listman.redhat.com/archives/libguestfs/2022-
February/msg00104.html>

=head1 SEE ALSO

L<libnbd(3)>.

--
Copyright (C) 2019-2022 Red Hat Inc.
```

errata-xmlrpc 2022-05-17 12:51:02 UTC Comment 13

```
Since the problem described in this bug report should be
resolved in a recent advisory, it has been closed with a
resolution of ERRATA.

For information on the advisory (new packages: libnbd), and
where to find the updated
files, follow the link below.

If the solution does not work for you, open a new bug report.
```

https://access.redhat.com/errata/RHEA-2022:2409

Note

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