```
commit 8e0969d06a58c7078fa5ad6f9a82e2595239b58f
Author: Greg Kroah-Hartman <gregkh@linuxfoundation.org>
Date:   Sat Mar 20 10:51:14 2021 +0100

    Linux 5.11.8

    Tested-by: Linux Kernel Functional Testing <lkft@linaro.org>
    Tested-by: Jason Self <jason@bluehome.net>
    Tested-by: Guenter Roeck <linux@roeck-us.net>
    Link: https://lore.kernel.org/r/20210319121747.203523570@linuxfoundation.org
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 8e9707717debce5b53b204520ed829d69cfc0162
Author: Ard Biesheuvel <ardb@kernel.org>
Date:   Thu Dec 31 17:41:54 2020 +0100

    crypto: x86/aes-ni-xts - use direct calls to and 4-way stride

    commit 86ad60a65f29dd862a11c22bb4b5be28d6c5cef1 upstream.

    The XTS asm helper arrangement is a bit odd: the 8-way stride helper
    consists of back-to-back calls to the 4-way core transforms, which
    are called indirectly, based on a boolean that indicates whether we
    are performing encryption or decryption.

    Given how costly indirect calls are on x86, let's switch to direct
    calls, and given how the 8-way stride doesn't really add anything
    substantial, use a 4-way stride instead, and make the asm core
    routine deal with any multiple of 4 blocks. Since 512 byte sectors
    or 4 KB blocks are the typical quantities XTS operates on, increase
    the stride exported to the glue helper to 512 bytes as well.

    As a result, the number of indirect calls is reduced from 3 per 64 bytes
    of in/output to 1 per 512 bytes of in/output, which produces a 65% speedup
    when operating on 1 KB blocks (measured on a Intel(R) Core(TM) i7-8650U CPU)

    Fixes: 9697fa39efd3f ("x86/retpoline/crypto: Convert crypto assembler indirect jumps")
    Tested-by: Eric Biggers <ebiggers@google.com> # x86_64
    Signed-off-by: Ard Biesheuvel <ardb@kernel.org>
    Signed-off-by: Herbert Xu <herbert@gondor.apana.org.au>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 6c5ebdff4ae0d64e34770c4d4793bff33c06035b
Author: Florian Fainelli <f.fainelli@gmail.com>
Date:   Mon Feb 22 14:30:10 2021 -0800

    net: dsa: b53: Support setting learning on port

    commit f9b3827ee66cfcf297d0acd6ecf33653a5f297ef upstream.

    Add support for being able to set the learning attribute on port, and
    make sure that the standalone ports start up with learning disabled.

    We can remove the code in bcm_sf2 that configured the ports learning
    attribute because we want the standalone ports to have learning disabled
    by default and port 7 cannot be bridged, so its learning attribute will
    not change past its initial configuration.

    Signed-off-by: Florian Fainelli <f.fainelli@gmail.com>
    Reviewed-by: Vladimir Oltean <olteanv@gmail.com>
    Signed-off-by: Jakub Kicinski <kuba@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit f8d4030a8a6787bd0aa5580c2f9dda4a6fafe1ec
```

```
Author: J. Bruce Fields <bfields@redhat.com>
Date:   Mon Mar 8 10:52:29 2021 -0500

    Revert "nfsd4: a client's own opens needn't prevent delegations"

    commit 6ee65a773096ab3f39d9b00311ac983be5bdeb7c upstream.

    This reverts commit 94415b06eb8aed13481646026dc995f04a3a534a.

    That commit claimed to allow a client to get a read delegation when it
    was the only writer.  Actually it allowed a client to get a read
    delegation when *any* client has a write open!

    The main problem is that it's depending on nfs4_clnt_odstate structures
    that are actually only maintained for pnfs exports.

    This causes clients to miss writes performed by other clients, even when
    there have been intervening closes and opens, violating close-to-open
    cache consistency.

    We can do this a different way, but first we should just revert this.

    I've added pynfs 4.1 test DELEG19 to test for this, as I should have
    done originally!

    Cc: stable@vger.kernel.org
    Reported-by: Timo Rothenpieler <timo@rothenpieler.org>
    Signed-off-by: J. Bruce Fields <bfields@redhat.com>
    Signed-off-by: Chuck Lever <chuck.lever@oracle.com>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit c9c48d3c779b970e11fd8fb279883bf92404865a
Author: J. Bruce Fields <bfields@redhat.com>
Date:   Mon Mar 8 10:51:51 2021 -0500

    Revert "nfsd4: remove check_conflicting_opens warning"

    commit 4aa5e002034f0701c3335379fd6c22d7f3338cce upstream.

    This reverts commit 50747dd5e47b "nfsd4: remove check_conflicting_opens
    warning", as a prerequisite for reverting 94415b06eb8a, which has a
    serious bug.

    Cc: stable@vger.kernel.org
    Signed-off-by: J. Bruce Fields <bfields@redhat.com>
    Signed-off-by: Chuck Lever <chuck.lever@oracle.com>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 5676df54d7d44f497b8dbf7bff04f2f1b165da93
Author: Amir Goldstein <amir73il@gmail.com>
Date:   Thu Mar 4 11:09:12 2021 +0200

    fuse: fix live lock in fuse_iget()

    commit 775c5033a0d164622d9d10dd0f0a5531639ed3ed upstream.

    Commit 5d069dbe8aaf ("fuse: fix bad inode") replaced make_bad_inode()
    in fuse_iget() with a private implementation fuse_make_bad().

    The private implementation fails to remove the bad inode from inode
    cache, so the retry loop with iget5_locked() finds the same bad inode
    and marks it bad forever.

    kmsg snip:
```

```
    [ ] rcu: INFO: rcu_sched self-detected stall on CPU
    ...
    [ ]  ? bit_wait_io+0x50/0x50
    [ ]  ? fuse_init_file_inode+0x70/0x70
    [ ]  ? find_inode.isra.32+0x60/0xb0
    [ ]  ? fuse_init_file_inode+0x70/0x70
    [ ]  ilookup5_nowait+0x65/0x90
    [ ]  ? fuse_init_file_inode+0x70/0x70
    [ ]  ilookup5.part.36+0x2e/0x80
    [ ]  ? fuse_init_file_inode+0x70/0x70
    [ ]  ? fuse_inode_eq+0x20/0x20
    [ ]  iget5_locked+0x21/0x80
    [ ]  ? fuse_inode_eq+0x20/0x20
    [ ]  fuse_iget+0x96/0x1b0

    Fixes: 5d069dbe8aaf ("fuse: fix bad inode")
    Cc: stable@vger.kernel.org # 5.10+
    Signed-off-by: Amir Goldstein <amir73il@gmail.com>
    Signed-off-by: Miklos Szeredi <mszeredi@redhat.com>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit e1a7ca048f05ae769ec4e9c50da06f2a3334872e
Author: Nicolas Morey-Chaisemartin <nmoreychaisemartin@suse.com>
Date:   Fri Feb 5 09:14:28 2021 +0100

    RDMA/srp: Fix support for unpopulated and unbalanced NUMA nodes

    commit 2b5715fc17386a6223490d5b8f08d031999b0c0b upstream.

    The current code computes a number of channels per SRP target and spreads
    them equally across all online NUMA nodes.  Each channel is then assigned
    a CPU within this node.

    In the case of unbalanced, or even unpopulated nodes, some channels do not
    get a CPU associated and thus do not get connected.  This causes the SRP
    connection to fail.

    This patch solves the issue by rewriting channel computation and
    allocation:

    - Drop channel to node/CPU association as it had no real effect on
      locality but added unnecessary complexity.

    - Tweak the number of channels allocated to reduce CPU contention when
      possible:
      - Up to one channel per CPU (instead of up to 4 by node)
      - At least 4 channels per node, unless ch_count module parameter is
        used.

    Link: https://lore.kernel.org/r/9cb4d9d3-30ad-2276-7eff-e85f7ddfb411@suse.com
    Signed-off-by: Nicolas Morey-Chaisemartin <nmoreychaisemartin@suse.com>
    Reviewed-by: Bart Van Assche <bvanassche@acm.org>
    Signed-off-by: Jason Gunthorpe <jgg@nvidia.com>
    Cc: Yi Zhang <yi.zhang@redhat.com>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 167a6f9124114f5d188a70b867f932d12745a444
Author: Vladimir Murzin <vladimir.murzin@arm.com>
Date:   Tue Mar 16 13:43:19 2021 +0000

    arm64: Unconditionally set virtual cpu id registers

    Commit 78869f0f0552 ("arm64: Extract parts of el2_setup into a macro")
    reorganized el2 setup in such way that virtual cpu id registers set
    only in nVHE, yet they used (and need) to be set irrespective VHE
```

```
    support.

    Fixes: 78869f0f0552 ("arm64: Extract parts of el2_setup into a macro")
    Signed-off-by: Vladimir Murzin <vladimir.murzin@arm.com>
    Acked-by: Will Deacon <will@kernel.org>
    Reviewed-by: Marc Zyngier <maz@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit bd3cc4be01bf91cd99c68427789df03cb6762868
Author: Piotr Krysiuk <piotras@gmail.com>
Date:   Tue Mar 16 11:44:42 2021 +0100

    bpf, selftests: Fix up some test_verifier cases for unprivileged

    commit 0a13e3537ea67452d549a6a80da3776d6b7dedb3 upstream.

    Fix up test_verifier error messages for the case where the original error
    message changed, or for the case where pointer alu errors differ between
    privileged and unprivileged tests. Also, add alternative tests for keeping
    coverage of the original verifier rejection error message (fp alu), and
    newly reject map_ptr += rX where rX == 0 given we now forbid alu on these
    types for unprivileged. All test_verifier cases pass after the change. The
    test case fixups were kept separate to ease backporting of core changes.

    Signed-off-by: Piotr Krysiuk <piotras@gmail.com>
    Co-developed-by: Daniel Borkmann <daniel@iogearbox.net>
    Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
    Acked-by: Alexei Starovoitov <ast@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit d94b5b83f42d50bd847f848b56d3a2b70ec7004b
Author: Piotr Krysiuk <piotras@gmail.com>
Date:   Tue Mar 16 09:47:02 2021 +0100

    bpf: Add sanity check for upper ptr_limit

    commit 1b1597e64e1a610c7a96710fc4717158e98a08b3 upstream.

    Given we know the max possible value of ptr_limit at the time of retrieving
    the latter, add basic assertions, so that the verifier can bail out if
    anything looks odd and reject the program. Nothing triggered this so far,
    but it also does not hurt to have these.

    Signed-off-by: Piotr Krysiuk <piotras@gmail.com>
    Co-developed-by: Daniel Borkmann <daniel@iogearbox.net>
    Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
    Acked-by: Alexei Starovoitov <ast@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 6bc7314a685952608e32e8776a39173130540d7f
Author: Piotr Krysiuk <piotras@gmail.com>
Date:   Tue Mar 16 08:26:25 2021 +0100

    bpf: Simplify alu_limit masking for pointer arithmetic

    commit b5871dca250cd391885218b99cc015aca1a51aea upstream.

    Instead of having the mov32 with aux->alu_limit - 1 immediate, move this
    operation to retrieve_ptr_limit() instead to simplify the logic and to
    allow for subsequent sanity boundary checks inside retrieve_ptr_limit().
    This avoids in future that at the time of the verifier masking rewrite
    we'd run into an underflow which would not sign extend due to the nature
    of mov32 instruction.

    Signed-off-by: Piotr Krysiuk <piotras@gmail.com>
```

```
    Co-developed-by: Daniel Borkmann <daniel@iogearbox.net>
    Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
    Acked-by: Alexei Starovoitov <ast@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit c4f3aa4343deccf5b8e1bfcc7c36224aaf3a8b26
Author: Piotr Krysiuk <piotras@gmail.com>
Date:   Tue Mar 16 08:20:16 2021 +0100

    bpf: Fix off-by-one for area size in creating mask to left

    commit 10d2bb2e6b1d8c4576c56a748f697dbeb8388899 upstream.

    retrieve_ptr_limit() computes the ptr_limit for registers with stack and
    map_value type. ptr_limit is the size of the memory area that is still
    valid / in-bounds from the point of the current position and direction
    of the operation (add / sub). This size will later be used for masking
    the operation such that attempting out-of-bounds access in the speculative
    domain is redirected to remain within the bounds of the current map value.

    When masking to the right the size is correct, however, when masking to
    the left, the size is off-by-one which would lead to an incorrect mask
    and thus incorrect arithmetic operation in the non-speculative domain.
    Piotr found that if the resulting alu_limit value is zero, then the
    BPF_MOV32_IMM() from the fixup_bpf_calls() rewrite will end up loading
    0xffffffff into AX instead of sign-extending to the full 64 bit range,
    and as a result, this allows abuse for executing speculatively out-of-
    bounds loads against 4GB window of address space and thus extracting the
    contents of kernel memory via side-channel.

    Fixes: 979d63d50c0c ("bpf: prevent out of bounds speculation on pointer arithmetic")
    Signed-off-by: Piotr Krysiuk <piotras@gmail.com>
    Co-developed-by: Daniel Borkmann <daniel@iogearbox.net>
    Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
    Acked-by: Alexei Starovoitov <ast@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 6bf7609666f6b2a9169c39c79a47ef8d6082afae
Author: Piotr Krysiuk <piotras@gmail.com>
Date:   Tue Mar 16 09:47:02 2021 +0100

    bpf: Prohibit alu ops for pointer types not defining ptr_limit

    commit f232326f6966cf2a1d1db7bc917a4ce5f9f55f76 upstream.

    The purpose of this patch is to streamline error propagation and in particular
    to propagate retrieve_ptr_limit() errors for pointer types that are not defining
    a ptr_limit such that register-based alu ops against these types can be rejected.

    The main rationale is that a gap has been identified by Piotr in the existing
    protection against speculatively out-of-bounds loads, for example, in case of
    ctx pointers, unprivileged programs can still perform pointer arithmetic. This
    can be abused to execute speculatively out-of-bounds loads without restrictions
    and thus extract contents of kernel memory.

    Fix this by rejecting unprivileged programs that attempt any pointer arithmetic
    on unprotected pointer types. The two affected ones are pointer to ctx as well
    as pointer to map. Field access to a modified ctx' pointer is rejected at a
    later point in time in the verifier, and 7c6967326267 ("bpf: Permit map_ptr
    arithmetic with opcode add and offset 0") only relevant for root-only use cases.
    Risk of unprivileged program breakage is considered very low.

    Fixes: 7c6967326267 ("bpf: Permit map_ptr arithmetic with opcode add and offset 0")
    Fixes: b2157399cc98 ("bpf: prevent out-of-bounds speculation")
    Signed-off-by: Piotr Krysiuk <piotras@gmail.com>
```

```
    Co-developed-by: Daniel Borkmann <daniel@iogearbox.net>
    Signed-off-by: Daniel Borkmann <daniel@iogearbox.net>
    Acked-by: Alexei Starovoitov <ast@kernel.org>
    Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

commit 4da985773cbb92cbf9659bdddf9789d699b5bfc6
Author: Bob Peterson <rpeterso@redhat.com>
Date:   Fri Mar 12 07:58:54 2021 -0500

    gfs2: bypass signal_our_withdraw if no journal

    [ Upstream commit d5bf630f355d8c532bef2347cf90e8ae60a5f1bd ]

    Before this patch, function signal_our_withdraw referenced the journal
    inode immediately. But corrupt file systems may have some invalid
    journals, in which case our attempt to read it in will withdraw and the
    resulting signal_our_withdraw would dereference the NULL value.

    This patch adds a check to signal_our_withdraw so that if the journal
    has not yet been initialized, it simply returns and does the old-style
    withdraw.

    Thanks, Andy Price, for his analysis.

    Reported-by: syzbot+50a8a9cf8127f2c6f5df@syzkaller.appspotmail.com
    Fixes: 601ef0d52e96 ("gfs2: Force withdraw to replay journals and wait for it to
finish")
    Signed-off-by: Bob Peterson <rpeterso@redhat.com>
    Signed-off-by: Andreas Gruenbacher <agruenba@redhat.com>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 0b150ae395ce1b73867981efc7537fd092ca3de1
Author: Bob Peterson <rpeterso@redhat.com>
Date:   Tue Dec 22 14:43:28 2020 -0600

    gfs2: move freeze glock outside the make_fs_rw and _ro functions

    [ Upstream commit 96b1454f2e8ede4c619fde405a1bb4e9ba8d218e ]

    Before this patch, sister functions gfs2_make_fs_rw and gfs2_make_fs_ro locked
    (held) the freeze glock by calling gfs2_freeze_lock and gfs2_freeze_unlock.
    The problem is, not all the callers of gfs2_make_fs_ro should be doing this.
    The three callers of gfs2_make_fs_ro are: remount (gfs2_reconfigure),
    signal_our_withdraw, and unmount (gfs2_put_super). But when unmounting the
    file system we can get into the following circular lock dependency:

    deactivate_super
       down_write(&s->s_umount); <------------------------------------ s_umount
       deactivate_locked_super
          gfs2_kill_sb
             kill_block_super
                generic_shutdown_super
                   gfs2_put_super
                      gfs2_make_fs_ro
                         gfs2_glock_nq_init sd_freeze_gl
                            freeze_go_sync
                               if (freeze glock in SH)
                                  freeze_super (vfs)
                                     down_write(&sb->s_umount); <------- s_umount

    This patch moves the hold of the freeze glock outside the two sister rw/ro
    functions to their callers, but it doesn't request the glock from
    gfs2_put_super, thus eliminating the circular dependency.

    Signed-off-by: Bob Peterson <rpeterso@redhat.com>
```

```
    Signed-off-by: Andreas Gruenbacher <agruenba@redhat.com>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit d21188031bf2d1e11d08159dc8262930723c2a09
Author: Bob Peterson <rpeterso@redhat.com>
Date:   Tue Dec 22 14:43:27 2020 -0600

    gfs2: Add common helper for holding and releasing the freeze glock

    [ Upstream commit c77b52c0a137994ad796f44544c802b0b766e496 ]

    Many places in the gfs2 code queued and dequeued the freeze glock.
    Almost all of them acquire it in SHARED mode, and need to specify the
    same LM_FLAG_NOEXP and GL_EXACT flags.

    This patch adds common helper functions gfs2_freeze_lock and gfs2_freeze_unlock
    to make the code more readable, and to prepare for the next patch.

    Signed-off-by: Bob Peterson <rpeterso@redhat.com>
    Signed-off-by: Andreas Gruenbacher <agruenba@redhat.com>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 7bd6bd2a9e7d386c40e431b0b399dc35ec157d89
Author: Frieder Schrempf <frieder.schrempf@kontron.de>
Date:   Mon Feb 22 12:52:20 2021 +0100

    regulator: pca9450: Clear PRESET_EN bit to fix BUCK1/2/3 voltage setting

    [ Upstream commit 98b94b6e38ca0c4eeb29949c656f6a315000c23e ]

    The driver uses the DVS registers PCA9450_REG_BUCKxOUT_DVS0 to set the
    voltage for the buck regulators 1, 2 and 3. This has no effect as the
    PRESET_EN bit is set by default and therefore the preset values are used
    instead, which are set to 850 mV.

    To fix this we clear the PRESET_EN bit at time of initialization.

    Fixes: 0935ff5f1f0a ("regulator: pca9450: add pca9450 pmic driver")
    Cc: <stable@vger.kernel.org>
    Signed-off-by: Frieder Schrempf <frieder.schrempf@kontron.de>
    Link: https://lore.kernel.org/r/20210222115229.166620-1-frieder.schrempf@kontron.de
    Signed-off-by: Mark Brown <broonie@kernel.org>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 22b6cb4fb1a60e6f52e1270052f4384324943777
Author: Frieder Schrempf <frieder.schrempf@kontron.de>
Date:   Thu Feb 11 11:55:30 2021 +0100

    regulator: pca9450: Enable system reset on WDOG_B assertion

    [ Upstream commit f7684f5a048febd2a7bc98ee81d6dce52f7268b8 ]

    By default the PCA9450 doesn't handle the assertion of the WDOG_B
    signal, but this is required to guarantee that things like software
    resets triggered by the watchdog work reliably.

    As we don't want to rely on the bootloader to enable this, we tell
    the PMIC to issue a cold reset in case the WDOG_B signal is
    asserted (WDOG_B_CFG = 10), just as the NXP U-Boot code does.

    Signed-off-by: Frieder Schrempf <frieder.schrempf@kontron.de>
    Link: https://lore.kernel.org/r/20210211105534.38972-3-frieder.schrempf@kontron.de
    Signed-off-by: Mark Brown <broonie@kernel.org>
    Signed-off-by: Sasha Levin <sashal@kernel.org>
```

```
commit 06d05cf7216c4a478775c7ab3a847aa8c8d3e1cd
Author: Frieder Schrempf <frieder.schrempf@kontron.de>
Date:   Thu Feb 11 11:55:28 2021 +0100

    regulator: pca9450: Add SD_VSEL GPIO for LDO5

    [ Upstream commit 8c67a11bae889f51fe5054364c3c789dfae3ad73 ]

    LDO5 has two separate control registers. LDO5CTRL_L is used if the
    input signal SD_VSEL is low and LDO5CTRL_H if it is high.
    The current driver implementation only uses LDO5CTRL_H. To make this
    work on boards that have SD_VSEL connected to a GPIO, we add support
    for specifying an optional GPIO and setting it to high at probe time.

    In the future we might also want to add support for boards that have
    SD_VSEL set to a fixed low level. In this case we need to change the
    driver to be able to use the LDO5CTRL_L register.

    Signed-off-by: Frieder Schrempf <frieder.schrempf@kontron.de>
    Link: https://lore.kernel.org/r/20210211105534.38972-1-frieder.schrempf@kontron.de
    Signed-off-by: Mark Brown <broonie@kernel.org>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 9b7ba5ae234d8b4838e649cc67c6adbe9b388d98
Author: Jens Axboe <axboe@kernel.dk>
Date:   Thu Mar 4 21:02:58 2021 -0700

    io_uring: clear IOCB_WAITQ for non -EIOCBQUEUED return

    [ Upstream commit b5b0ecb736f1ce1e68eb50613c0cfecff10198eb ]

    The callback can only be armed, if we get -EIOCBQUEUED returned. It's
    important that we clear the WAITQ bit for other cases, otherwise we can
    queue for async retry and filemap will assume that we're armed and
    return -EAGAIN instead of just blocking for the IO.

    Cc: stable@vger.kernel.org # 5.9+
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 308f41d96630aeca838b4f4bce8056a0f35ea64c
Author: Pavel Begunkov <asml.silence@gmail.com>
Date:   Mon Feb 1 18:59:56 2021 +0000

    io_uring: simplify do_read return parsing

    [ Upstream commit 57cd657b8272a66277c139e7bbdc8b86057cb415 ]

    do_read() returning 0 bytes read (not -EAGAIN/etc.) is not an important
    enough of a case to prioritise it. Fold it into ret < 0 check, so we get
    rid of an extra if and make it a bit more readable.

    Signed-off-by: Pavel Begunkov <asml.silence@gmail.com>
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit ee8869b7a07181d24bb4fd355a4482b17621d409
Author: Jens Axboe <axboe@kernel.dk>
Date:   Thu Mar 4 17:15:48 2021 -0700

    io_uring: don't keep looping for more events if we can't flush overflow

    [ Upstream commit ca0a26511c679a797f86589894a4523db36d833e ]

    It doesn't make sense to wait for more events to come in, if we can't
```

```
    even flush the overflow we already have to the ring. Return -EBUSY for
    that condition, just like we do for attempts to submit with overflow
    pending.

    Cc: stable@vger.kernel.org # 5.11
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 7eae0ee0eb3c16e1bfeaa3dd1cbd3617a5c766ac
Author: Pavel Begunkov <asml.silence@gmail.com>
Date:   Thu Feb 4 13:51:58 2021 +0000

    io_uring: refactor io_cqring_wait

    [ Upstream commit eeb60b9ab4000d20261973642dfc9fb0e4b5d073 ]

    It's easy to make a mistake in io_cqring_wait() because for all
    break/continue clauses we need to watch for prepare/finish_wait to be
    used correctly. Extract all those into a new helper
    io_cqring_wait_schedule(), and transforming the loop into simple series
    of func calls: prepare(); check_and_schedule(); finish();

    Signed-off-by: Pavel Begunkov <asml.silence@gmail.com>
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit f6acca687580deb89a1a24a53d8b2fba104668de
Author: Pavel Begunkov <asml.silence@gmail.com>
Date:   Thu Feb 4 13:51:57 2021 +0000

    io_uring: refactor scheduling in io_cqring_wait

    [ Upstream commit c1d5a224683b333ddbe278e455d639ccd4f5ca2b ]

    schedule_timeout() with timeout=MAX_SCHEDULE_TIMEOUT is guaranteed to
    work just as schedule(), so instead of hand-coding it based on arguments
    always use the timeout version and simplify code.

    Signed-off-by: Pavel Begunkov <asml.silence@gmail.com>
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit bda646bb954fb0974bff508819e87ce6783c35b1
Author: Florian Westphal <fw@strlen.de>
Date:   Thu Mar 4 13:32:11 2021 -0800

    mptcp: dispose initial struct socket when its subflow is closed

    [ Upstream commit 17aee05dc8822e354f5ad2d68ee39e3ba4b6acf2 ]

    Christoph Paasch reported following crash:
    dst_release underflow
    WARNING: CPU: 0 PID: 1319 at net/core/dst.c:175 dst_release+0xc1/0xd0 net/core/dst.c:175
    CPU: 0 PID: 1319 Comm: syz-executor217 Not tainted 5.11.0-
rc6af8e85128b4d0d24083c5cac646e891227052e0c #70
    Call Trace:
     rt_cache_route+0x12e/0x140 net/ipv4/route.c:1503
     rt_set_nexthop.constprop.0+0x1fc/0x590 net/ipv4/route.c:1612
     __mkroute_output net/ipv4/route.c:2484 [inline]
    ...

    The worker leaves msk->subflow alone even when it
    happened to close the subflow ssk associated with it.

    Fixes: 866f26f2a9c33b ("mptcp: always graft subflow socket to parent")
```

```
    Closes: https://github.com/multipath-tcp/mptcp_net-next/issues/157
    Reported-by: Christoph Paasch <cpaasch@apple.com>
    Suggested-by: Paolo Abeni <pabeni@redhat.com>
    Acked-by: Paolo Abeni <pabeni@redhat.com>
    Signed-off-by: Florian Westphal <fw@strlen.de>
    Signed-off-by: Mat Martineau <mathew.j.martineau@linux.intel.com>
    Signed-off-by: David S. Miller <davem@davemloft.net>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit a7171755c6cdccb23063e1c4de3f23210342f71f
Author: Florian Westphal <fw@strlen.de>
Date:   Thu Feb 4 15:23:30 2021 -0800

    mptcp: pm: add lockdep assertions

    [ Upstream commit 3abc05d9ef6fe989706b679e1e6371d6360d3db4 ]

    Add a few assertions to make sure functions are called with the needed
    locks held.
    Two functions gain might_sleep annotations because they contain
    conditional calls to functions that sleep.

    Signed-off-by: Florian Westphal <fw@strlen.de>
    Signed-off-by: Mat Martineau <mathew.j.martineau@linux.intel.com>
    Signed-off-by: Jakub Kicinski <kuba@kernel.org>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 00f7d503e29229ee08e6d8efccd9c70a15b4902d
Author: Geliang Tang <geliangtang@gmail.com>
Date:   Mon Feb 1 15:09:09 2021 -0800

    mptcp: send ack for every add_addr

    [ Upstream commit b5a7acd3bd63c7430c98d7f66d0aa457c9ccde30 ]

    This patch changes the sending ACK conditions for the ADD_ADDR, send an
    ACK packet for any ADD_ADDR, not just when ipv6 addresses or port
    numbers are included.

    Closes: https://github.com/multipath-tcp/mptcp_net-next/issues/139
    Acked-by: Paolo Abeni <pabeni@redhat.com>
    Signed-off-by: Geliang Tang <geliangtang@gmail.com>
    Signed-off-by: Mat Martineau <mathew.j.martineau@linux.intel.com>
    Signed-off-by: Jakub Kicinski <kuba@kernel.org>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit fae013c419c17ab5a1dac7d95e97c594e20cca82
Author: Sean Christopherson <seanjc@google.com>
Date:   Thu Feb 25 12:47:26 2021 -0800

    KVM: x86/mmu: Set SPTE_AD_WRPROT_ONLY_MASK if and only if PML is enabled

    [ Upstream commit 44ac5958a6c1fd91ac8810fbb37194e377d78db5 ]

    Check that PML is actually enabled before setting the mask to force a
    SPTE to be write-protected.  The bits used for the !AD_ENABLED case are
    in the upper half of the SPTE.  With 64-bit paging and EPT, these bits
    are ignored, but with 32-bit PAE paging they are reserved.  Setting them
    for L2 SPTEs without checking PML breaks NPT on 32-bit KVM.

    Fixes: 1f4e5fc83a42 ("KVM: x86: fix nested guest live migration with PML")
    Cc: stable@vger.kernel.org
    Signed-off-by: Sean Christopherson <seanjc@google.com>
    Message-Id: <20210225204749.1512652-2-seanjc@google.com>
    Signed-off-by: Paolo Bonzini <pbonzini@redhat.com>
```

```
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 2c23de8cbf13adbf5f5c3e4997b2c21ec571e49b
Author: Sean Christopherson <seanjc@google.com>
Date:   Fri Feb 12 16:50:08 2021 -0800

    KVM: x86/mmu: Expand on the comment in kvm_vcpu_ad_need_write_protect()

    [ Upstream commit 2855f98265dc579bd2becb79ce0156d08e0df813 ]

    Expand the comment about need to use write-protection for nested EPT
    when PML is enabled to clarify that the tagging is a nop when PML is
    _not_ enabled.  Without the clarification, omitting the PML check looks
    wrong at first^Wfifth glance.

    Signed-off-by: Sean Christopherson <seanjc@google.com>
    Message-Id: <20210213005015.1651772-8-seanjc@google.com>
    Signed-off-by: Paolo Bonzini <pbonzini@redhat.com>
    Signed-off-by: Sasha Levin <sashal@kernel.org>

commit 62c8121d230a849670d98f13af0bb1e74e8b9ea8
Author: Jens Axboe <axboe@kernel.dk>
Date:   Tue Feb 23 19:17:35 2021 -0700

    io_uring: don't attempt IO reissue from the ring exit path

    [ Upstream commit 7c977a58dc83366e488c217fd88b1469d242bee5 ]

    If we're exiting the ring, just let the IO fail with -EAGAIN as nobody
    will care anyway. It's not the right context to reissue from.

    Cc: stable@vger.kernel.org
    Signed-off-by: Jens Axboe <axboe@kernel.dk>
    Signed-off-by: Sasha Levin <sashal@kernel.org>
```