Conversions Content Policy

For 2026-05-06 PUCK

Contact ackerleytng@google.com if you have questions/suggestions!

In-place conversions v5 [1] posted

- Conversions == guest_memfd ioctl to set shared/private state

[1] RFC v5: https://lore.kernel.org/all/20260428-gmem-inplace-conversion-v5-0-d8608ccfca22@google.com/T/

Content Policy: Current

- Content policy: what userspace expects of the memory content after conversion
  - `ZERO`: When userspace next reads the converted memory, it will read zeros
  - `PRESERVE`: If host writes `0xbeef`, guest will read `0xbeef` after conversion
    - Applies in both directions
  - `MODE_UNSPECIFIED` (default): No guarantees.

Awkward

- Content policy: what userspace expects of the memory content after conversion
  - `ZERO`: When userspace next reads the converted memory, it will read zeros
    - => Contract doesn't extend to the guest - don't want to guarantee what guest sees
  - `PRESERVE`: If host writes `0xbeef`, guest will read `0xbeef` after conversion
    - => Contract extends to guest
  - `MODE_UNSPECIFIED` (default): No guarantees.
- => Too many special cases
  - `PRESERVE` only supported before finalization and for TDX, only for to_private conversions
  - SNP needs `PRESERVE` to_shared pre-finalization

Smaller contract

- Current: what userspace expects of the memory content after conversion
- New: what to expect of the memory content after guest_memfd updates attributes
  - No guarantees on what guest reads whatsoever: that is the contract between the trusted firmware and guest
    - pKVM - firmware guarantees guest no encryption

Implementation

- PRESERVE == guarantee that the process of setting memory attributes doesn't change memory contents.
  - Implementation == do nothing in most cases, except -EOPNOTSUPP for to-shared on TDX
- ZERO == guarantee that the process of setting memory attributes zeroes memory contents.
  - Implementation == memset(zero) in most cases.
- UNSPECIFIED == no guarantees
  - Implementation == guest_memfd does nothing explicitly about memory contents.
    - Pretty much the same as PRESERVE except guest_memfd won't take into account vendor-specific side effects of the process of setting memory attributes