ONLINE DISK REENCRYPTION

WITH LUKS2

ONDŘEJ KOZINA <OKOZINA@REDHAT.COM>

Devconf 2019 - Brno

WHY?

- Different data lifetime and algorithm lifetime

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)
  - LUKS passphrase change does not affect volume key (data encryption key)

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)
  - LUKS passphrase change does not affect volume key (data encryption key)
- Volume key change may be enforced by policy

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)
  - LUKS passphrase change does not affect volume key (data encryption key)
- Volume key change may be enforced by policy
- Offline cryptsetup-reencrypt misses few features

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)
  - LUKS passphrase change does not affect volume key (data encryption key)
- Volume key change may be enforced by policy
- Offline cryptsetup-reencrypt misses few features
  - not online

WHY?

- Different data lifetime and algorithm lifetime
- Cut-off access to data with volume key backup (LUKS header backup)
  - LUKS passphrase change does not affect volume key (data encryption key)
- Volume key change may be enforced by policy
- Offline cryptsetup-reencrypt misses few features
  - not online
  - not robust enough in case of failure

redhat

RESILIENT ONLINE

LINE DISK REENCRYPTION

WITH LUKS2

ONDŘEJ KOZINA <OKOZINA@REDHAT.COM>

Devconf 2019 - Brno

RESILIENT...?

- Reencryption writes device in segments (reencryption zone)
- Crash may produce torn write
- To detect and correct torn write we require:
  - Storage to keep additional information
  - Metadata format capable properly track reencryption progress
- LUKS2 format is flexible enough for it

BETTER TO BE SAFE...

- Checksums
  - 1 checksum per underlying physical sector size of underlying device
  - Single extra checksum of new ciphertext
  - Stored in keyslots binary area

BETTER TO BE SAFE...

- Checksums
  - 1 checksum per underlying physical sector size of underlying device
  - Single extra checksum of new ciphertext
  - Stored in keyslots binary area
- Journal

BETTER TO BE SAFE...

- Checksums
  - 1 checksum per underlying physical sector size of underlying device
  - Single extra checksum of new ciphertext
  - Stored in keyslots binary area
- Journal
- Data shift

BETTER TO BE SAFE...

- Checksums
  - 1 checksum per underlying physical sector size of underlying device
  - Single extra checksum of new ciphertext
  - Stored in keyslots binary area
- Journal
- Data shift
- Noop
  - No syncs
  - No commit points
  - Only gracefully interrupted reencryption is safe

IT CRASHED!

- Checksums
  - Compare content of reencryption zone to stored checksums
  - Reencrypt only sectors with matching checksums
  - Verify new ciphertext segment matches final checksum

IT CRASHED!

- Checksums
  - Compare content of reencryption zone to stored checksums
  - Reencrypt only sectors with matching checksums
  - Verify new ciphertext segment matches final checksum
- Journal
  - Replay

IT CRASHED!

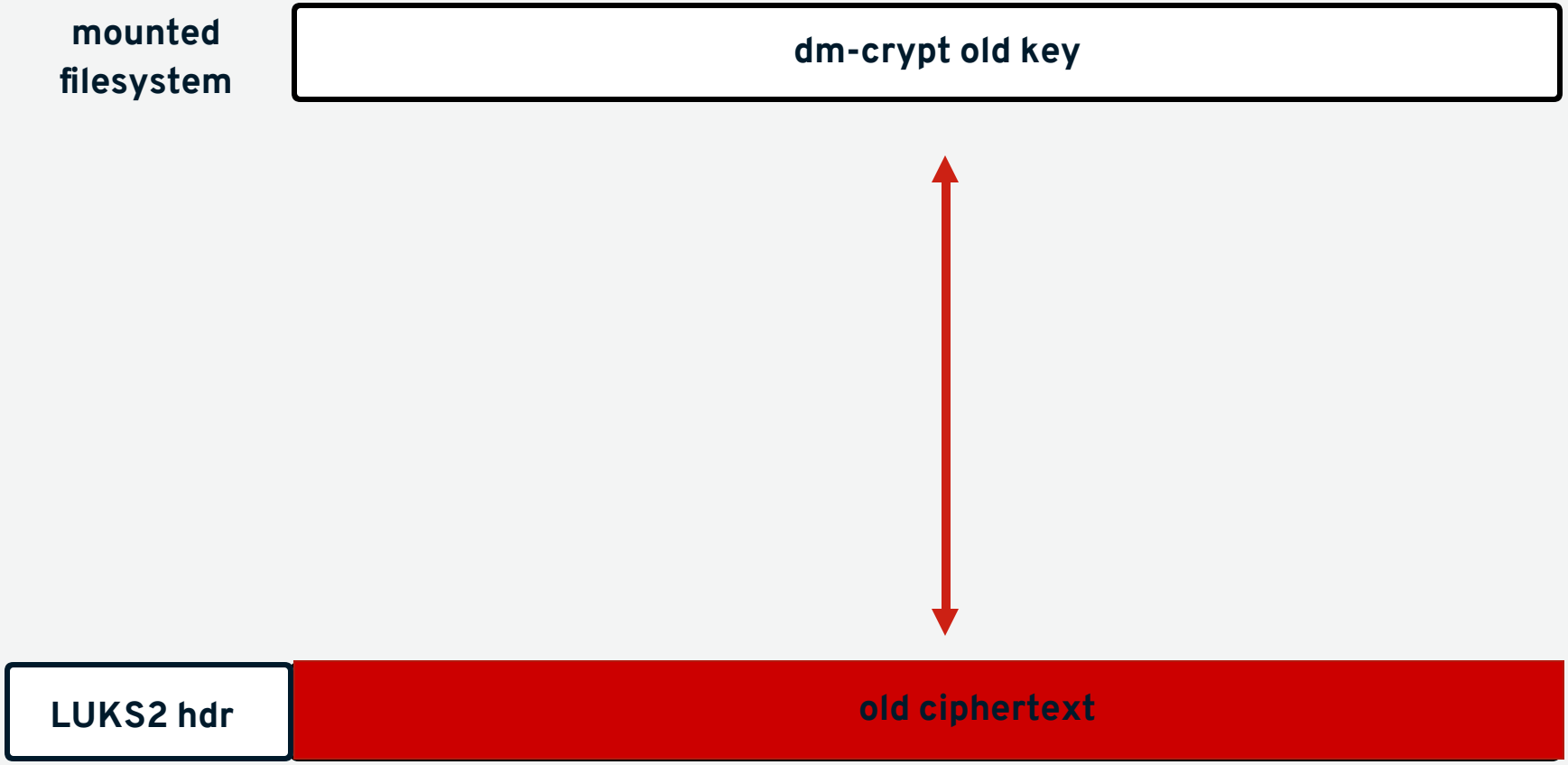
- Checksums
  - Compare content of reencryption zone to stored checksums
  - Reencrypt only sectors with matching checksums
  - Verify new ciphertext segment matches final checksum
- Journal
  - Replay
- Data shift
  - Repeat

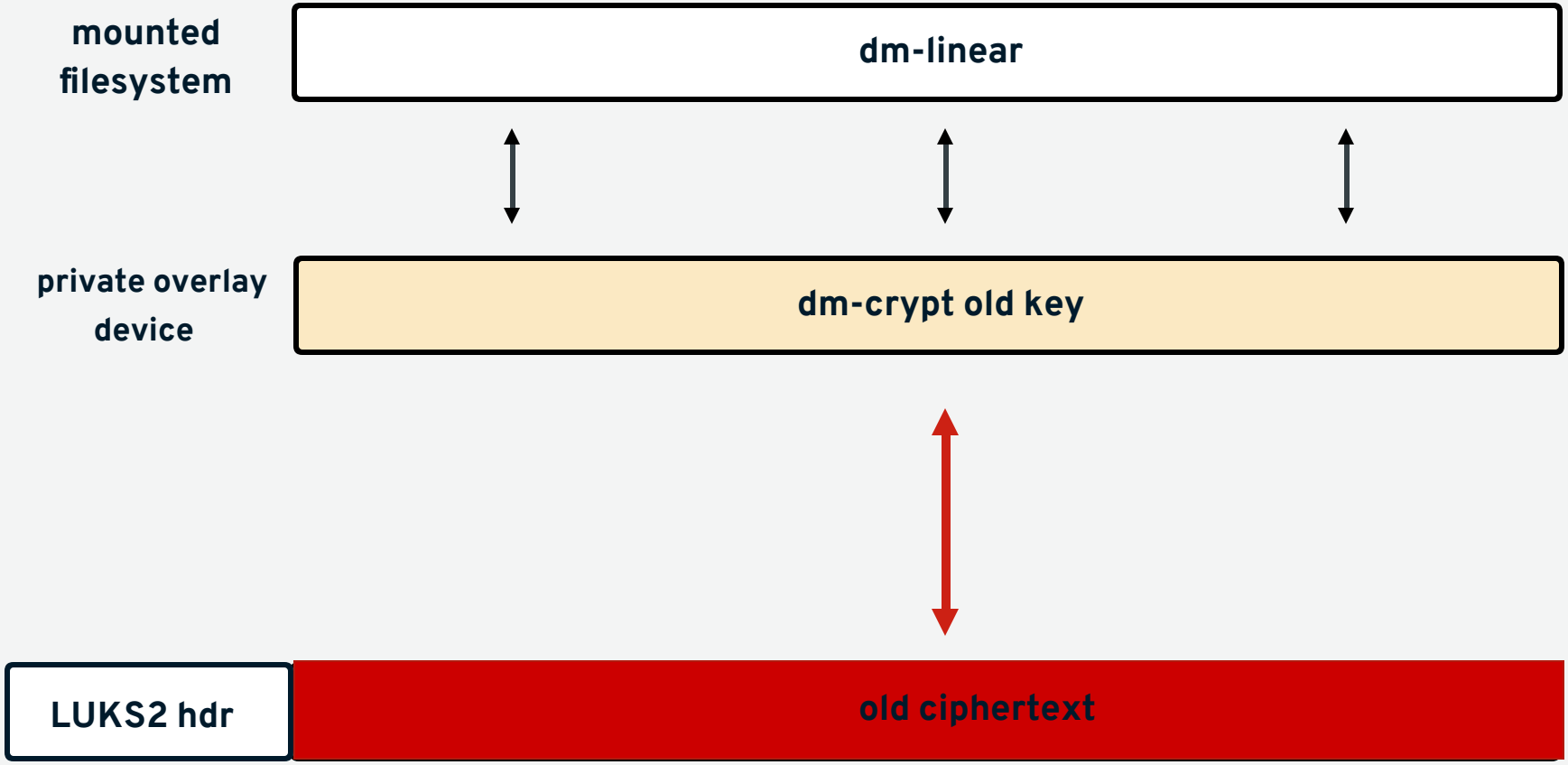
IT CRASHED!

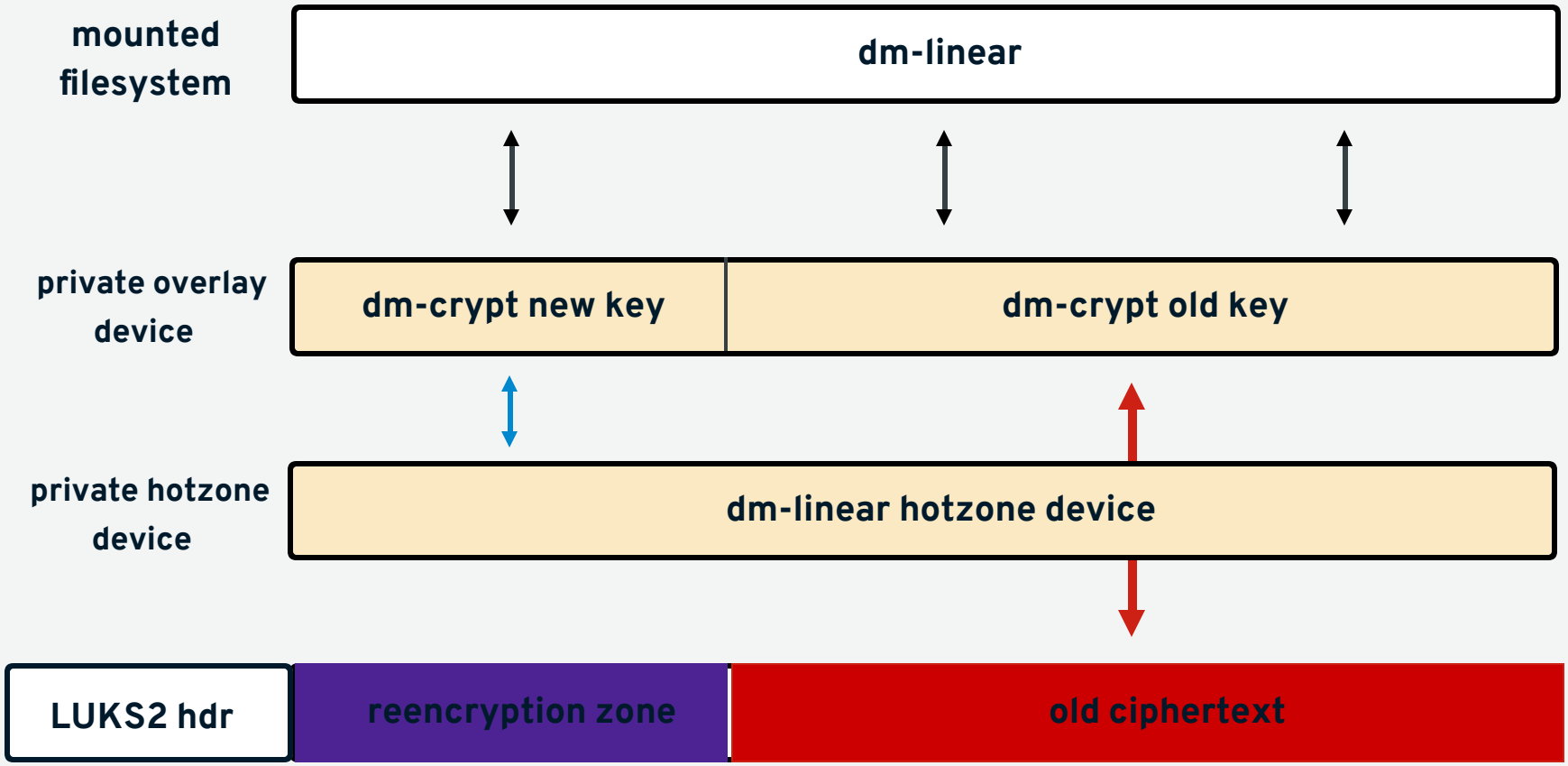
- Checksums
  - Compare content of reencryption zone to stored checksums
  - Reencrypt only sectors with matching checksums
  - Verify new ciphertext segment matches final checksum
- Journal
  - Replay
- Data shift
  - Repeat
- Noop
  - :(

ONLINE REENCRYPTION

- Independet layer
- Filesystem/applications access the data via device-mapper stack
- Reencryption process controls access to reencryption zone

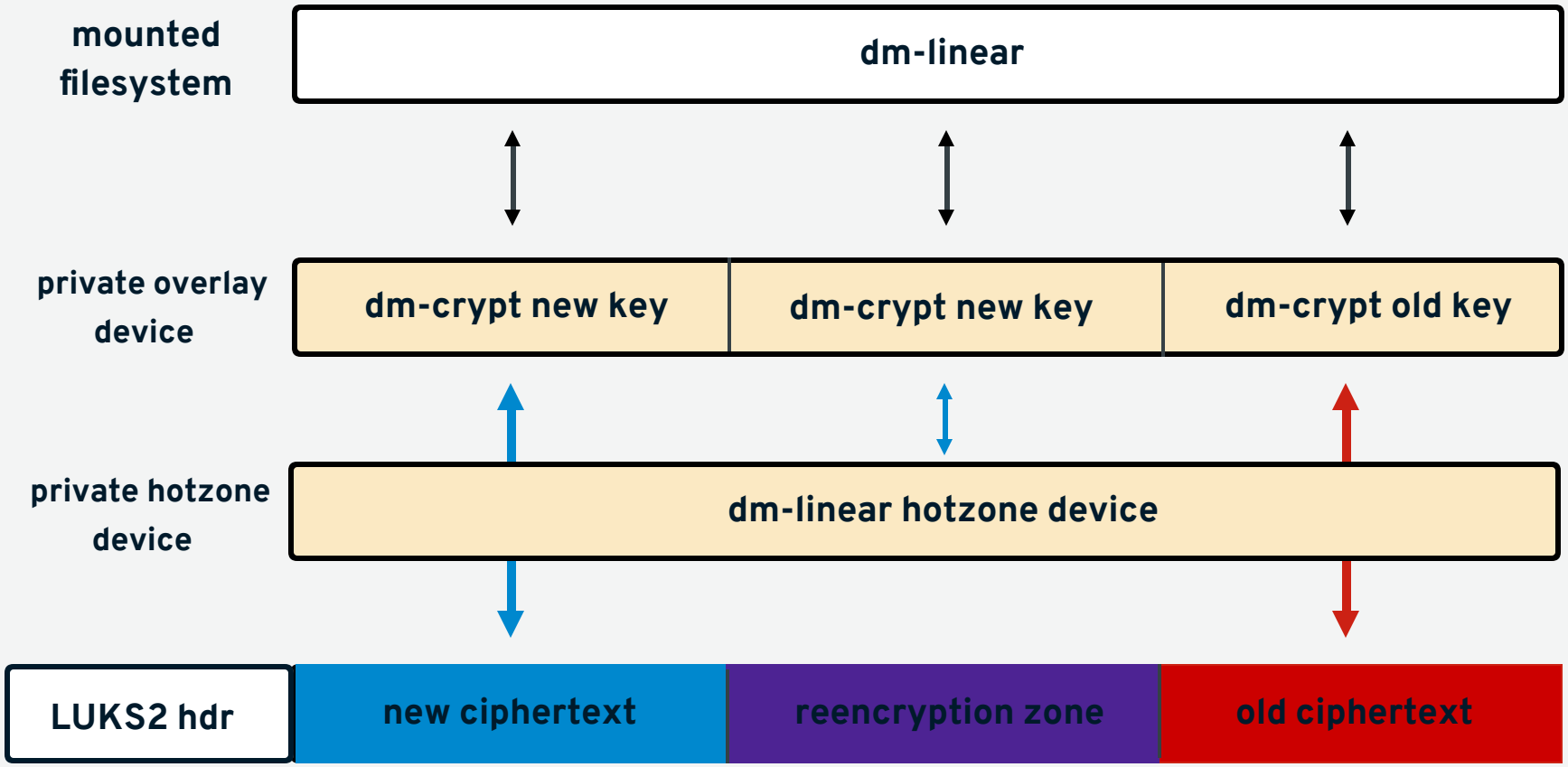

mounted filesystem
dm-crypt old key
LUKS2 hdr
old ciphertext


mounted filesystem
dm-linear
private overlay device
dm-crypt old key
LUKS2 hdr
old ciphertext



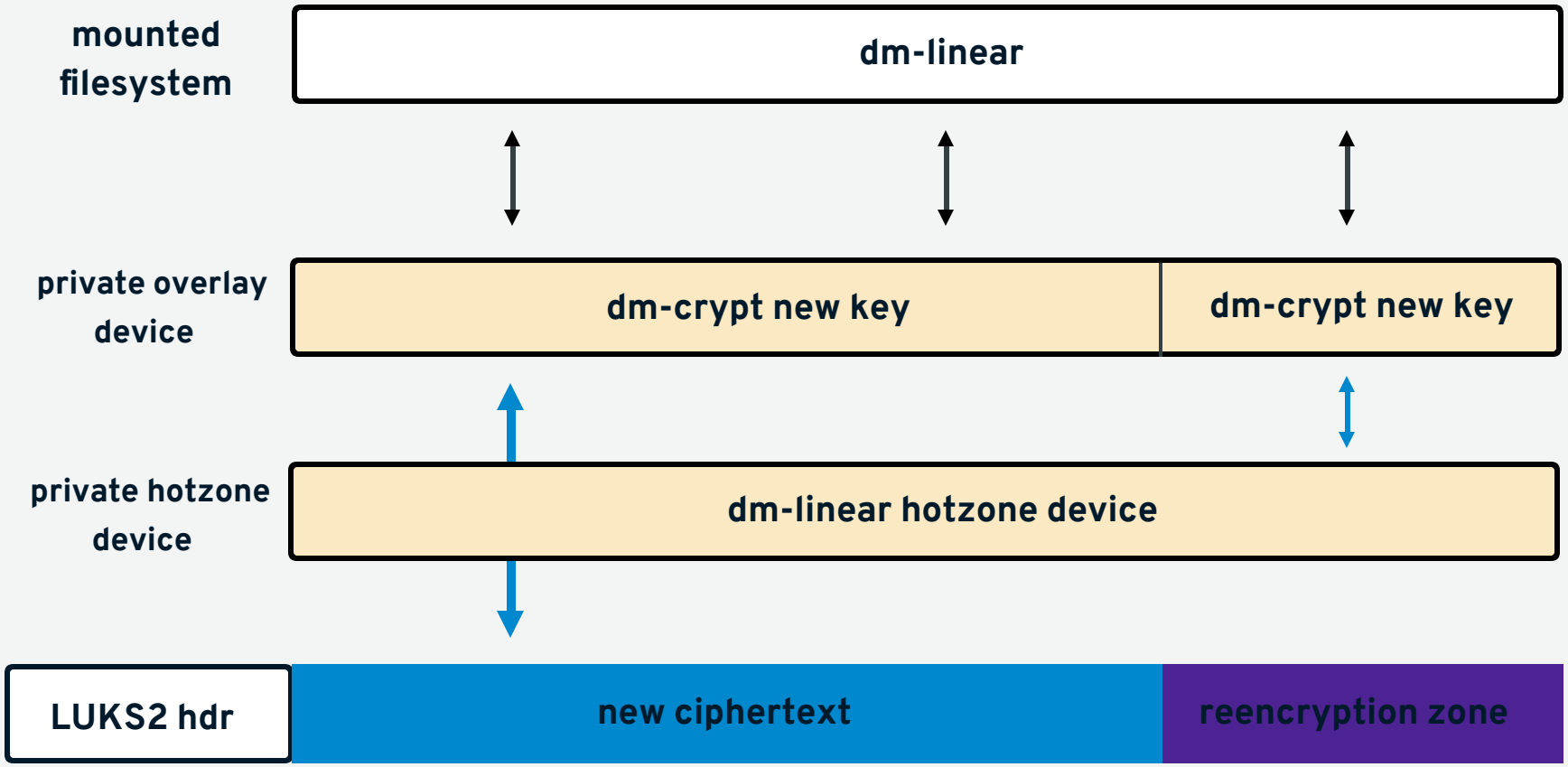
FOR EACH REENCRYPTION ZONE (CHECKSUMS):

1. Read old ciphertext
2. Calculate and store checkums in LUKS2 metadata (commit point)
3. Write new ciphertext
4. Update metadata (commit point)



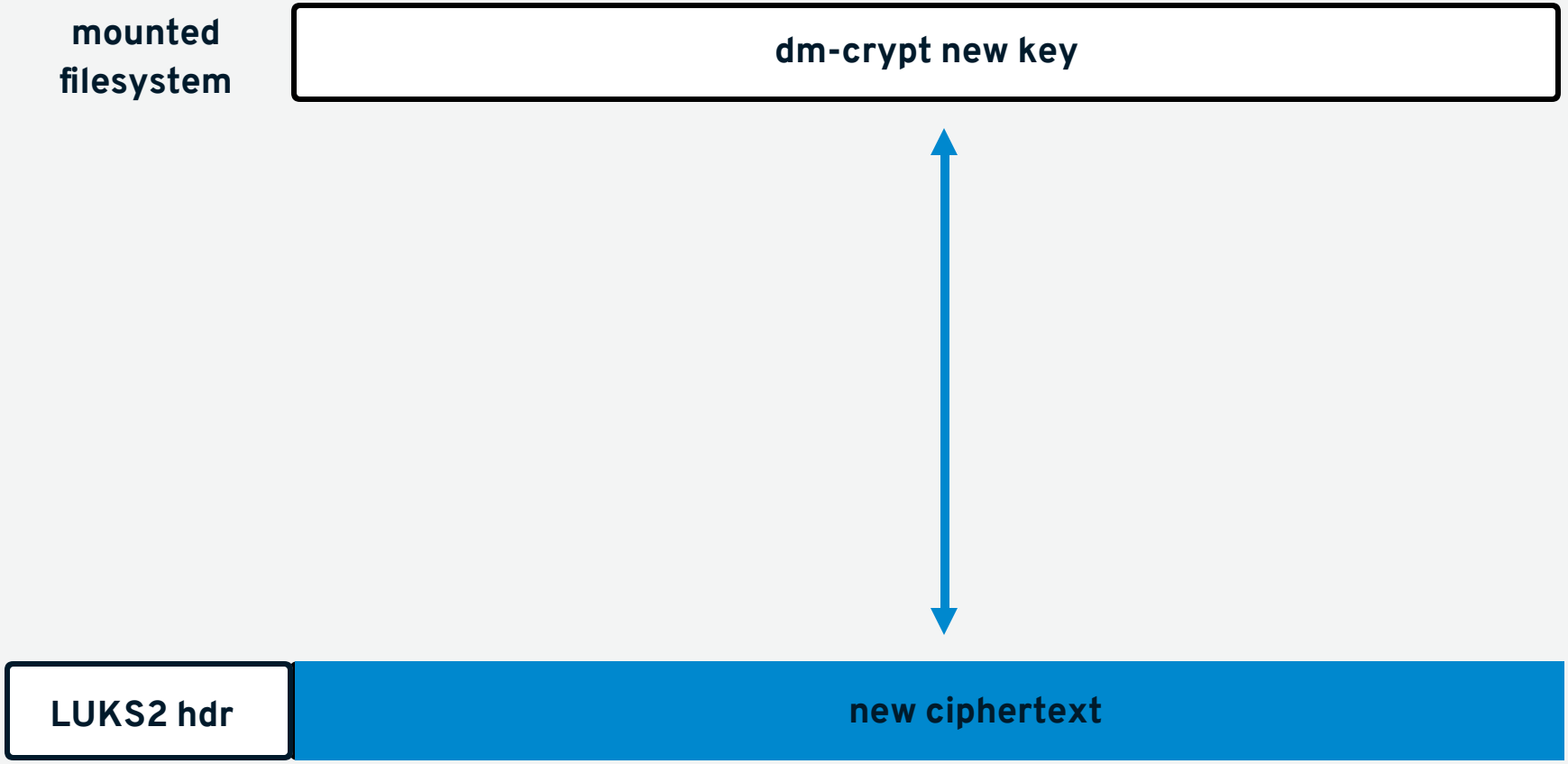
FOR EACH REENCRYPTION ZONE (CHECKSUMS):

1. Read old ciphertext
2. Calculate and store checkums in LUKS2 metadata (commit point)
3. Write new ciphertext
4. Update metadata (commit point)



FOR EACH REENCRYPTION ZONE (CHECKSUMS):

1. Read old ciphertext
2. Calculate and store checkums in LUKS2 metadata (commit point)
3. Write new ciphertext
4. Update metadata (commit point)


mounted filesystem
dm-crypt new key
LUKS2 hdr
new ciphertext

PERFORMANCE BASELINE:

- sequential **read**
- sequential **write**
- random reads and writes, **randrw** (70% reads, 30% writes)
- fio utility (iodepth=16, 4 threads, block size 32KiB, direct io)

ROTATIONAL HDD (5400 RPMS, 512 GB)

storage	read	write	randrw
raw	125 MiB/s	94 MiB/s	read 8,6 MiB/s, write 3,6 MiB/s
dm-crypt	125 MiB/s	65 MiB/s	read 9,3 MiB/s, write 4,0 MiB/s

NVME (512 GB)

storage	read	write	randrw
raw	2,2 GiB/s	875 MiB/s	862 MiB/s, 369 MiB/s
dm-crypt	2,2 GiB/s	875MiB/s	833 MiB/s, 374 MiB/s

REENCRYPTION PERFORMANCE:

NVME

reencryption zone size	resilience mode	idle	ETA	load	ETA
95 MiB (3,7 MiB)	checksums	206 MiB/s	43m	156 MiB/s	56m
198 MiB (7,7 MiBs)	checksums	212 MiB/s	42m	175 MiB/s	50m
3,7 MiB	journal	116 MiB/s	1h49m	45 MiB/s	3h15m
7,7 MiB	journal	159 MiB/s	55m	67 MiB/s	2h11m
20 MiB	noop	400 MiB/s	22m	240 MiB/s	37m
95 MiB (detached hdr)	checksums	206 MiB/s	43m	156 MiB/s	56m
198 MiB (detached hdr)	checksums	212 MiB/s	42m	175 MiB/s	50m
8 MiB	data shift	235 MiB/s	38m	97 MiB/s	1h31m

ETA: estimated time to reencrypt 512 GiB device

REENCRYPTION PERFORMANCE:

ROTATIONAL HDD

reencryption zone size	resilience mode	idle	ETA	load	ETA
95 MiB (3,7 MiB)	checksums	37 MiB/s	4h	17 MiB/s	8h34m
198 MiB (7,7 MiB)	checksums	40 MiB/s	3h40m	29 MiB/s	5h2m
3,7 MiB	journal	11 MiB/s	13h15m	4 MiB/s	1d12h
7,7 MiB	journal	16 MiB/s	9h6m	6 MiB/s	1d
20 MiB	noop	50 MiB/s	2h55m	25 MiB/s	5h50m
95 MiB (detached hdr)	checksums	45 MiB/s	3h15m	33 MiB/s	4h25m
198 MiB (detached hdr)	checksums	45 MiB/s	3h15m	35 MiB/s	4h10m
8 MiB	data shift	15 MiB/s	9h43m	5 MiB/s	1d5h

ETA: estimated time to reencrypt 512 GiB device

SUMMARY

- Reencryption
- Encryption (short offline period)
- Decryption
  - currently detached LUKS2 header only
- Can interrupted, paused, resumed with different parameters
- Recovery performed in **crypt_activate_by_*** calls (cryptsetup open)

DEMO

THANK YOU!
Q&A

- https://gitlab.com/cryptsetup/cryptsetup
- contacts: Ondřej Kozina <okozina@redhat.com>