Beavers, dams and kernel development

KVM Forum 2024

Paolo Bonzini, Red Hat
Distinguished Engineer

--verbose

- Recap on confidential computing technologies
- What’s up with TDX upstreaming?
- How can distros help developing for TDX?

TDX vs. SEV-SNP

- Competing technologies for AMD and Intel
- Confidentiality and integrity on untrusted hosts
- Completely different implementation, same building blocks:
    - Memory encryption
    - Protected guest state
    - Protected page tables

Memory encryption

- Both add encryption to the memory controller
- Intel: MK/TME (Multi-Key Total Memory Encryption)
  - Not limited to virtualization
  - Key ID bits stored in the page table entries
  - Configured via PCONFIG instruction
- AMD: SEV (Secure Encrypted Virtualization)
  - Key ID associated with virtual machine ASID
  - Configured via Secure Processor

Protected guest state (SEV-ES, TDX)

- Hypervisor cannot capture/modify register state
  - Values in registers (e.g. keys!)
  - Replay attacks via EIP
- VMEXITs passed to guest exception handler if:
  - Synchronous with respect to guest execution
  - Access to execution state required
- Guest places subset of register state in unencrypted memory, issues VMGEXIT/TDVMCALL instruction

Protected page tables (SEV-SNP, TDX)

- Limits to GPA→HPA translation
  - No memory aliasing attacks via n:1 translation
  - No memory remapping attacks
- Tracking of page ownership
  - Hypervisor
  - Guest
  - System (e.g., VMSA)

SEV-SNP vs. TDX

- Host↔SP interface
- New instructions
  - RMPUPDATE
  - PVALIDATE
- Guest↔SP encrypted communication

- VMX usage mediated by a “lowvisor” (TDX module):
  - VMCS access
  - Building private EPT
  - VM entry, first dib at VM exit
- Special processor mode (SEAM) provides isolation
- Not unlike pKVM (in theory)

SEV upstreaming

- 4.14 (2017): Secure Memory Encryption (SME)
- 4.15 (2018): SEV guest support
- 4.16: SEV support in KVM
- 5.10 (2020): SEV-ES guest support
- 5.11: SEV-ES host support
- 5.19 (2022): SEV-SNP guest support
- 6.11 (2024): SEV-SNP host support

TDX upstreaming

- 5.19 (2022): TDX guest support
- 6.8 (2024): TDX module initialization
- ???

TDX issues

Other architectures beware...

- No chances of getting API wrong
- Really complex piece of software
  - Spec extremely difficult to change
  - Complex interactions between guest and host
- Large amount of code (130 patches, +8700/-700)
  - ... with many dependencies, too

1. Wait for KVM to support TDX

2. ???

3. PROFIT!

How do I work with TDX?

- Vendor kernel sources
- Alternative kernel packages from your distro

Problems

- Vendor kernel sources are a black box
  - Periodically rebased, but it’s not clear what enters and leaves
  - Not clear what has been posted upstream and what hasn’t
- The distros conundrum
  - Building packages from vendor kernels has the same issues
  - Curating the set of patches is a lot of work

Steve Hersey,
Wikimedia Commons,
CC-BY-SA 2.0

What about beavers?

- Beaver dams slow down rivers, preventing erosion
- Beaver dams create wetlands that stop wildfires
- Beavers create the environment in which other species thrive
- Beavers help the ecosystem

What about beavers?

- Beaver dams slow down rivers, preventing erosion
- Beaver dams create wetlands that stop wildfires
- Beavers create the environment in which other species thrive
- Beavers help the ecosystem

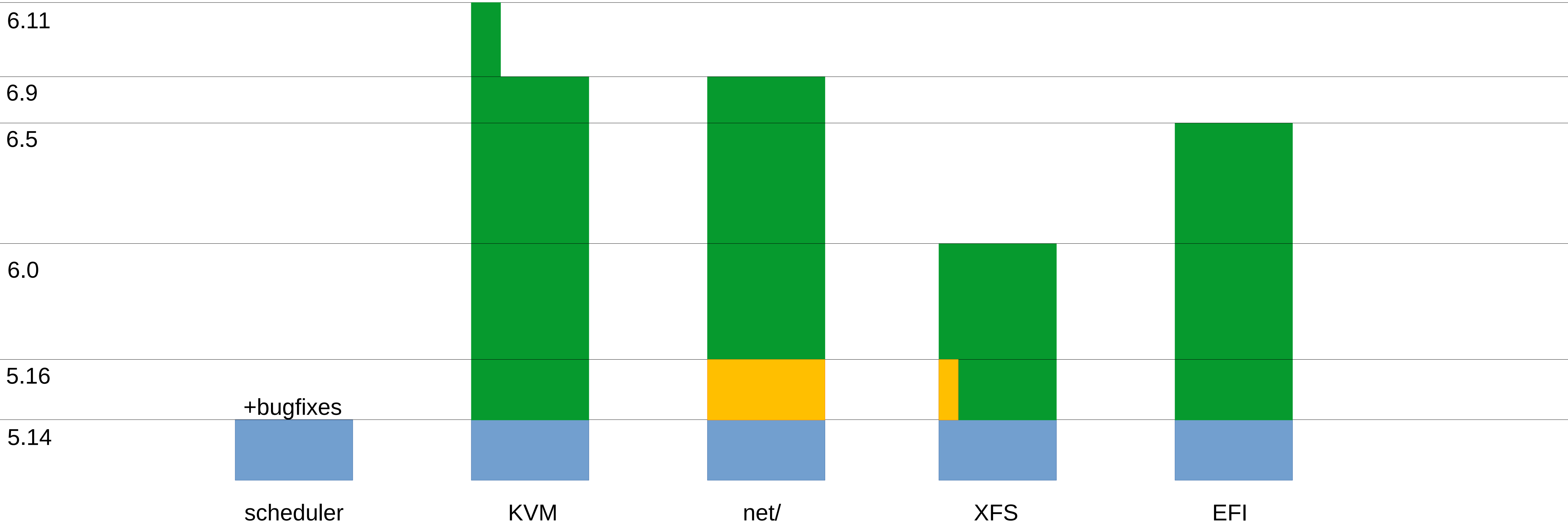
Two approaches to stable kernels

- Upstream LTS
- CentOS Stream kernel

Some numbers

- ~100,000 commits between 5.14 and RHEL 9.5
- 40% of upstream commits in the same period
- 15,000 for Linux 5.15 (LTS)
- 25,000 for Linux 4.14 (end-of-life)

The frankenkernel



What goes into the frankenkernel?

- Bugfixes (1-5 commits)
- Feature backports (10-50 commits)
- Mass backports (100+ commits)

Cycle of product development

- ~3 years of continuous development
- ~5 years of maintenance

	Stable kernels	CentOS Stream kernels
Development	New kernel versions	3 years full support
Maintenance	New LTS kernel releases	2 years full support 5 years maintenance (source only)

Selling points

- At least 3 people take a look at the list of patches
- All upstream Fixes must be included or waived
- Deviations from upstream are checked by hand
- Sanity testing (at least) for each merge request
- Source available on gitlab, packages via dnf or Koji (https://kojihub.stream.centos.org/koji/)

What makes this possible?

- Good upstream practices
  - Commit messages
  - Well-defined maintainer boundaries
  - Topic branches
- Downstream planning and infrastructure
  - CI
  - Bots

The CentOS TDX packages

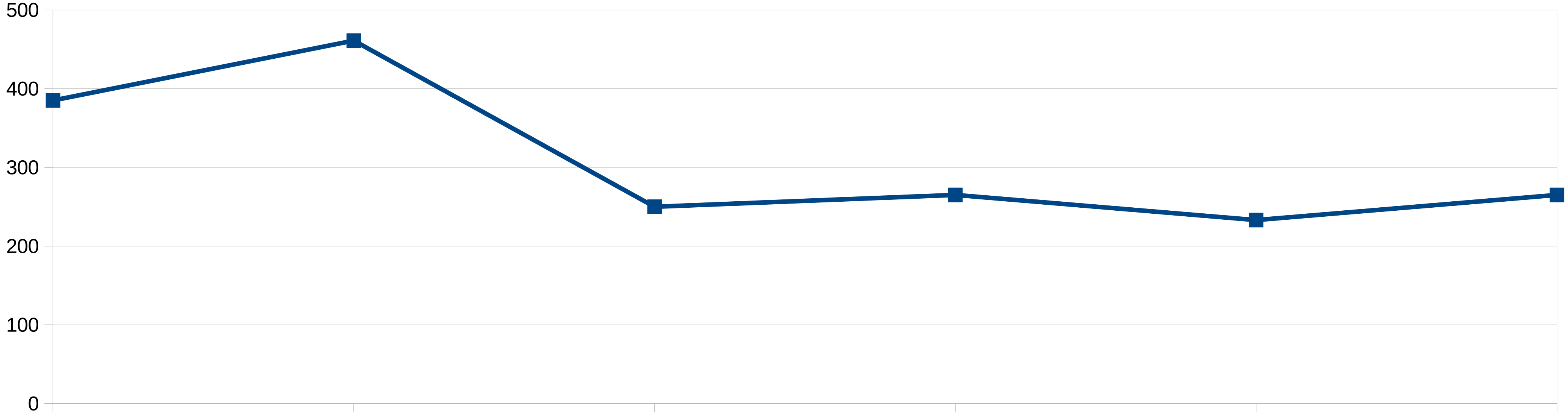
- Maintained by CentOS virtualization SIG (Paolo Bonzini, Camilla Conte, Cole Robinson)
- Based on the CentOS Stream kernel plus:
  - Recent (YMMV) patches for TDX support*
  - Possibly, upcoming merge requests for KVM or x86
- Same advantages, but on a smaller but moving target
- Integrated with the rest of the distribution

*currently based on v15

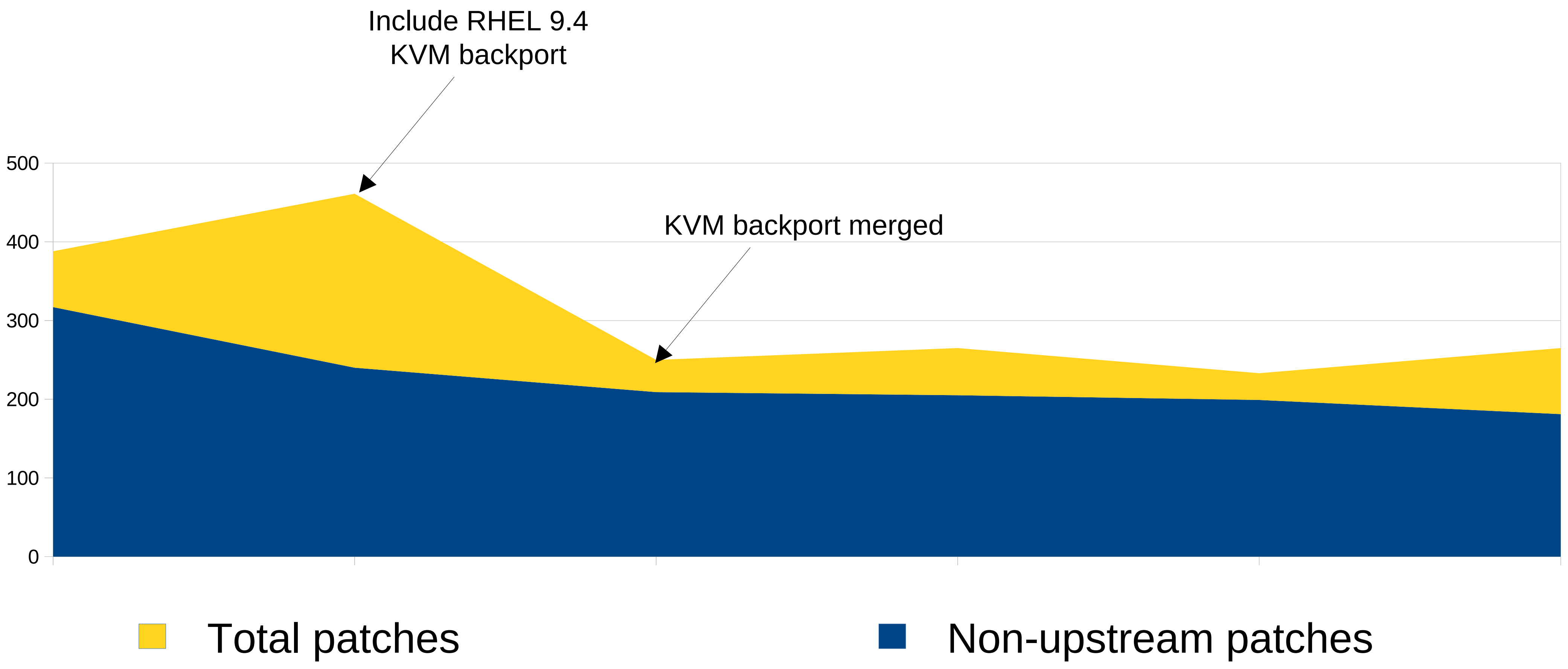
The CentOS TDX packages

- Generally faithful to mailing list posting
- Rebased to new upstream infrastructure
- Consistent API/ABI for kernel, QEMU and libvirt
- Source also available on gitlab, packages via dnf or Koji (https://cbs.centos.org)

The CentOS TDX packages



The CentOS TDX packages



What has CentOS ever done for us?

- Minimal set of downstream patches
  - Limited to what is in active discussion upstream
  - Updated to a reasonably recent kernel
- Non-upstream, non-posted patches identified and analyzed
- One source for kernel, QEMU and libvirt that work together

Questions

- linkedin.com/company/red-hat
- youtube.com/user/RedHatVideos
- facebook.com/redhatinc
- twitter.com/RedHat