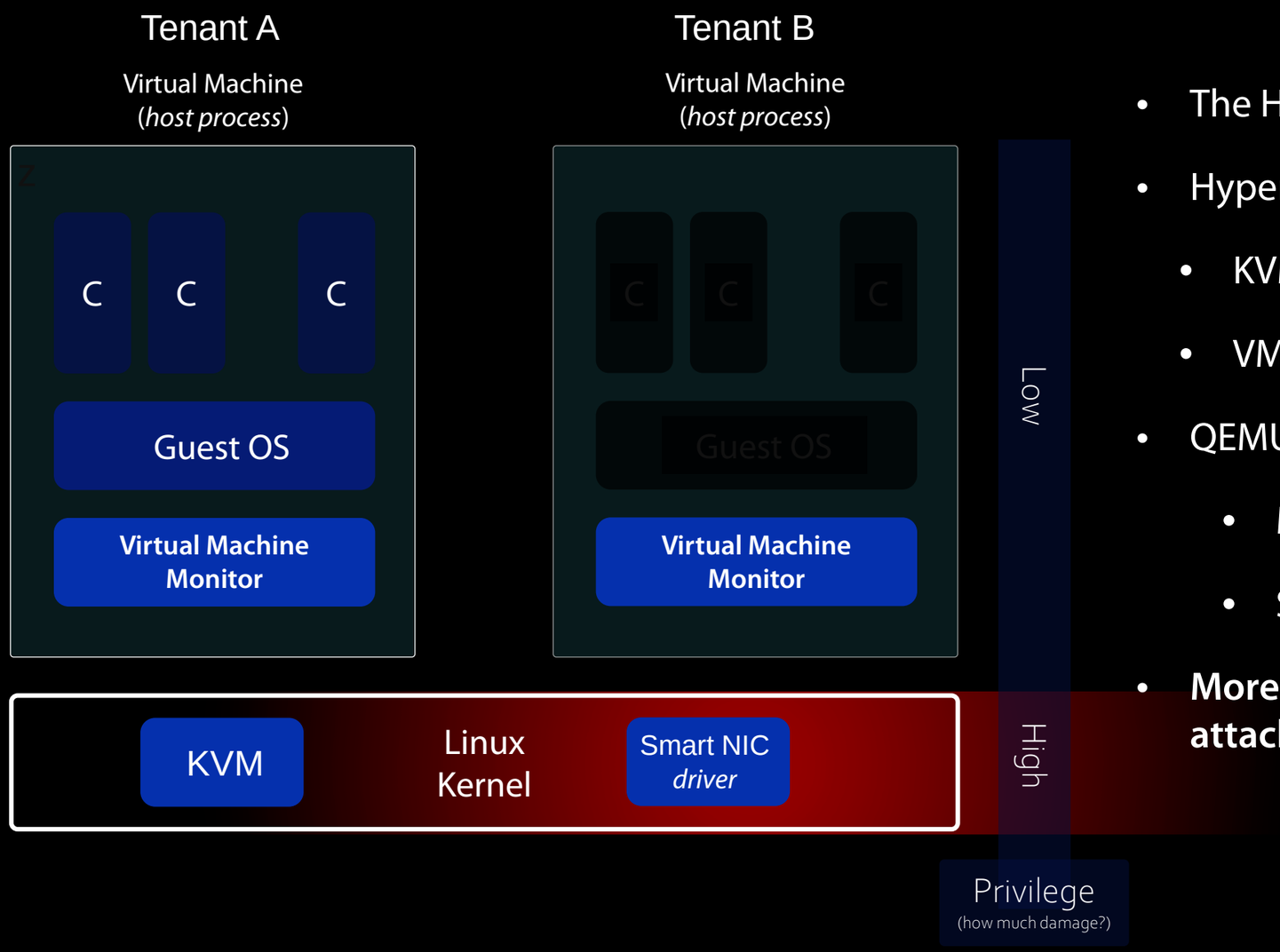
Improving Hypervisor Security

Team: **Daniele Buono**, Carlo Bertolli, Tobin Feldman-Fitzthum, Hubertus Franke, James Bottomley
IBM Research, T.J. Watson Research Center

The Hypervisor is a very powerful piece of software



- The Hypervisor is managing the (software) wall
- Hypervisor in Linux is split
  - KVM, with kernel-level privileges
  - VMM, with user-space privileges
- QEMU is the most complex VMM
  - Many of the features are actually not needed
  - Some may not be tested/maintained properly
- **More features -> more code -> larger surface of attack -> Less security**

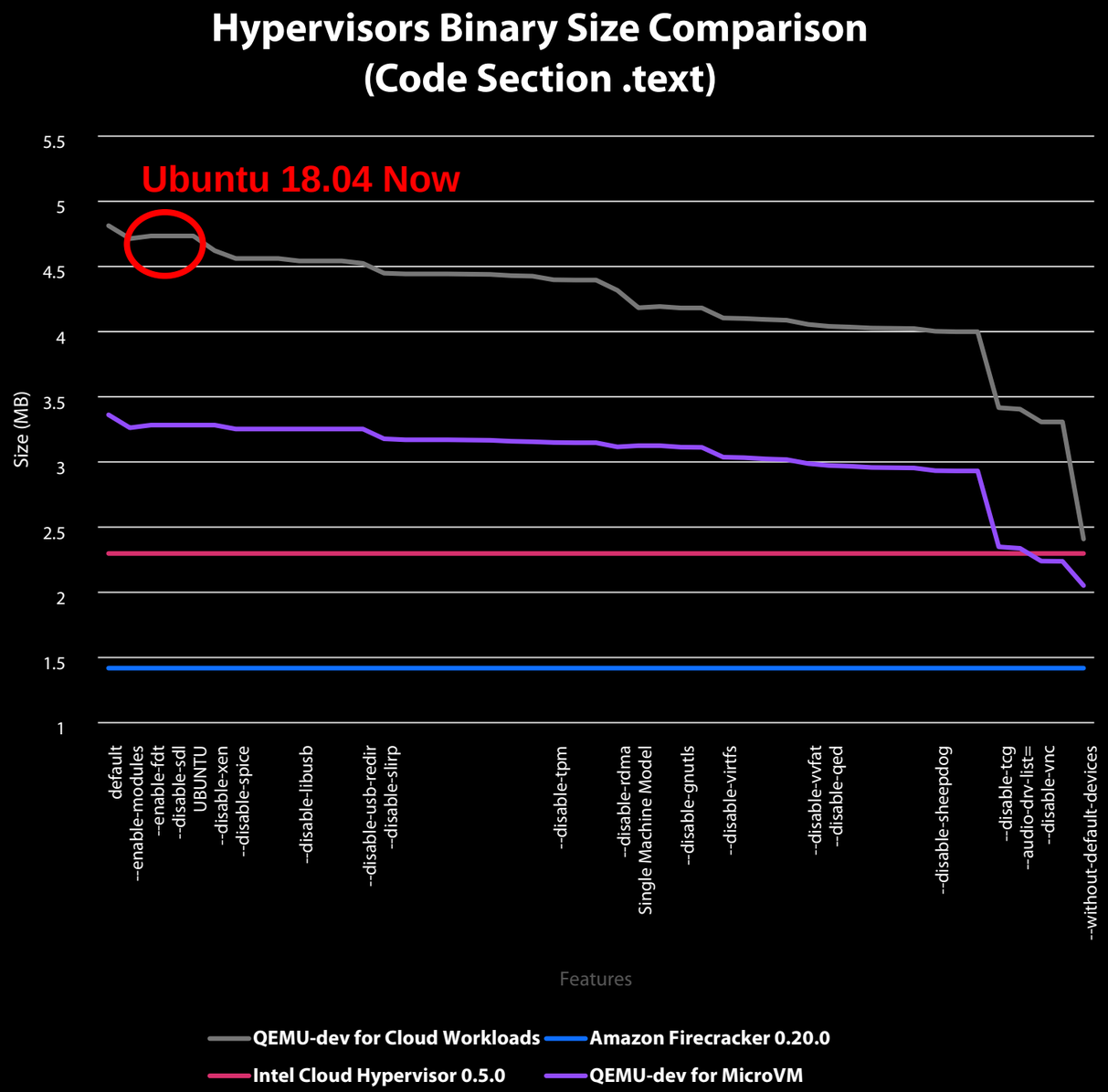
Measuring surface of attack via lines of code

- Since most of the features of QEMU are not needed for the specific case of Cloud Virtualization, why not remove them?
- QEMU is not modular

Measuring surface of attack via lines of code

- Since most of the features of QEMU are not needed for the specific case of Cloud Virtualization, why not remove them?
- ~~QEMU is not modular~~
  - QEMU has done a lot of progress in terms of modularity and code reduction capabilities in the past two years; however
  - Such changes have yet to reach most Linux distributions
    - Slow production cycles
    - Loss of generality

Measuring surface of attack via lines of code



- Since most of the features of QEMU are not needed for the specific case of Cloud Virtualization, why not remove them?
- ~~QEMU is not modular~~
  - QEMU has done a lot of progress in terms of modularity and code reduction capabilities in the past two years; however
  - Such changes have yet to reach most Linux distributions
    - Slow production cycles
    - Loss of generality
- Collaborating with QEMU maintainers, we were able to explore the current modularity of QEMU
  - The picture is very promising, with QEMU already a serious contender to Cloud Hypervisor

Improving Metrics in Security

Binary Size

Converging to

[2.5-4.5] MB

Qemu, Cloud Hypervisor

Defect Density

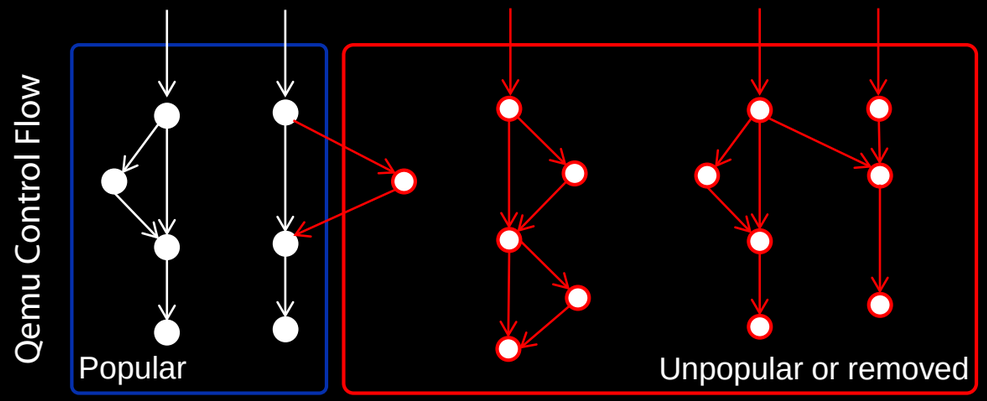
(https://scan.coverity.com/projects/qemu)

0.01 over ~2M LOCs

average Open Source project is 0.65

Bug Location in Code

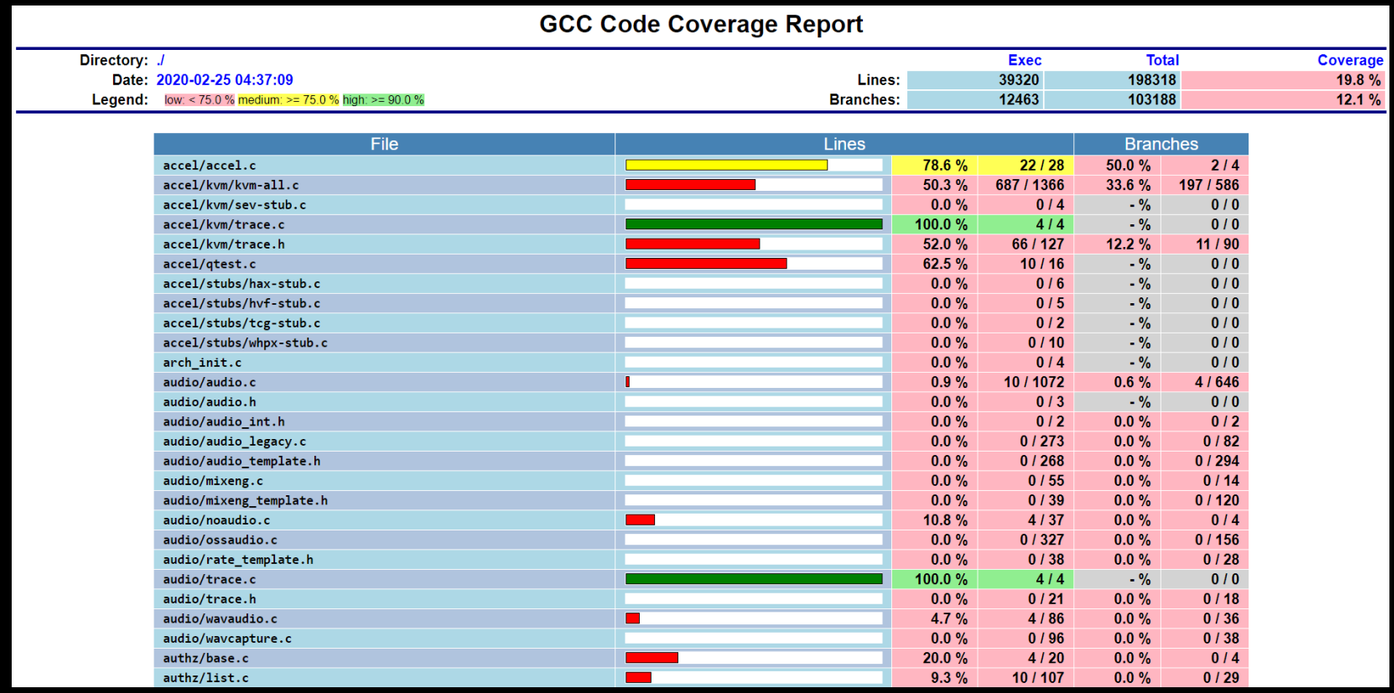
Coverage under common Cloud Workloads



CVE frequency in popular and unpopular/unused paths

Improving Metrics in Security - Bug Location in Code

- Location of fixed bugs through git commit log:
  - CVE-2019-12068
    - hw/scsi/lsi53c895a.c ✓ Code Removed
  - CVE-2018-7550
    - hw/i386/multiboot.c ? Code not removed
  - CVE-2018-5683
    - hw/display/vga.c ✓ Code Removed
  - CVE-2018-19489
    - hw/9pfs/9p.c ✓ Code Removed
  - CVE-2018-19364
    - hw/9pfs/9p.c ✓ Code Removed
  - CVE-2018-17963
    - net/net.c net/net.h ? Code not removed
  - CVE-2018-16872
    - hw/usb/dev-mtp.c ✓ Code Removed
  - CVE-2018-16867
    - hw/usb/dev-mtp.c ✓ Code Removed
  - CVE-2018-16847
    - hw/block/nvme.c ? Code not removed
  - CVE-2017-9503
    - hw/scsi/megasas.c ✓ Code Removed
  - CVE-2017-8379
    - ui/input.c ? Code not removed



GCC Code Coverage Report

Directory: ./
Date: 2020-02-25 04:37:09
Legend: low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %

	Exec	Total	Coverage
Lines:	39320	198318	19.8 %
Branches:	12463	103188	12.1 %

File	Lines		Branches	
accel/accel.c	78.6 %	22 / 28	50.0 %	2 / 4
accel/kvm/kvm-all.c	50.3 %	687 / 1366	33.6 %	197 / 586
accel/kvm/sev-stub.c	0.0 %	0 / 4	- %	0 / 0
accel/kvm/trace.c	100.0 %	4 / 4	- %	0 / 0
accel/kvm/trace.h	52.0 %	66 / 127	12.2 %	11 / 90
accel/qtest.c	62.5 %	10 / 16	- %	0 / 0
accel/stubs/hax-stub.c	0.0 %	0 / 6	- %	0 / 0
accel/stubs/hvf-stub.c	0.0 %	0 / 5	- %	0 / 0
accel/stubs/tcg-stub.c	0.0 %	0 / 2	- %	0 / 0
accel/stubs/whpx-stub.c	0.0 %	0 / 10	- %	0 / 0
arch_init.c	0.0 %	0 / 4	- %	0 / 0
audio/audio.c	0.9 %	10 / 1072	0.6 %	4 / 646
audio/audio.h	0.0 %	0 / 3	- %	0 / 0
audio/audio_int.h	0.0 %	0 / 2	0.0 %	0 / 2
audio/audio_legacy.c	0.0 %	0 / 273	0.0 %	0 / 82
audio/audio_template.h	0.0 %	0 / 268	0.0 %	0 / 294
audio/mixeng.c	0.0 %	0 / 55	0.0 %	0 / 14
audio/mixeng_template.h	0.0 %	0 / 39	0.0 %	0 / 120
audio/noaudio.c	10.8 %	4 / 37	0.0 %	0 / 4
audio/ossaudio.c	0.0 %	0 / 327	0.0 %	0 / 156
audio/rate_template.h	0.0 %	0 / 38	0.0 %	0 / 28
audio/trace.c	100.0 %	4 / 4	- %	0 / 0
audio/trace.h	0.0 %	0 / 21	0.0 %	0 / 18
audio/wavaudio.c	4.7 %	4 / 86	0.0 %	0 / 36
audio/wavcapture.c	0.0 %	0 / 96	0.0 %	0 / 38
authz/base.c	20.0 %	4 / 20	0.0 %	0 / 4
authz/list.c	9.3 %	10 / 107	0.0 %	0 / 29

- Next Step:
- Work on Coverage Reports
  - Isolate popular and unpopular paths
  - Isolate unreachable paths, such as:
    - devices not enabled at runtime
    - interfaces not accessible from the guest (e.g. monitor)
  - Correlate past CVEs to the popularity of the paths

Improving Hypervisor Security

- Current focus only on VMM -> QEMU
  - Code reduction
    - Latest QEMU versions have multiple features for specialization
  - Work on metrics to measure security
    - Coverage, code path classification
    - Bug Localization
      - How many of the bugs found in the past 4 years would have been avoided with specialization?
      - Are bugs more frequent in most used or least used parts of the code?