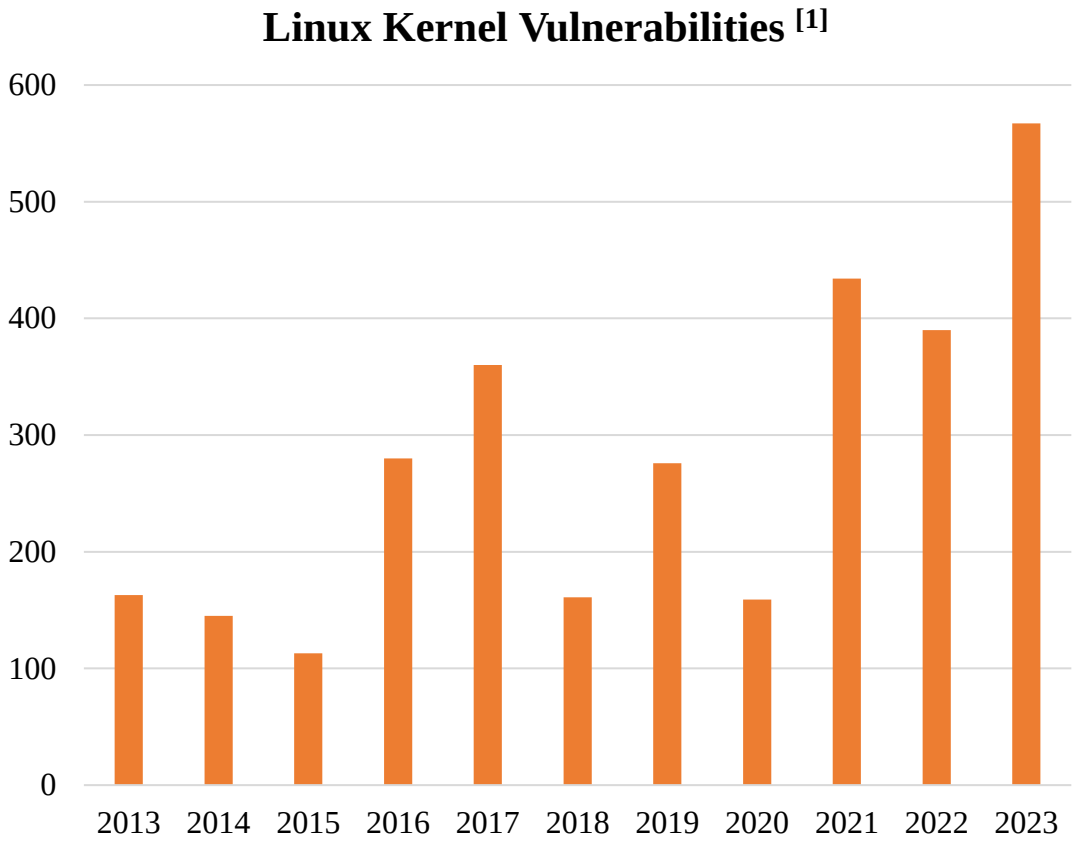
ISSTA 24

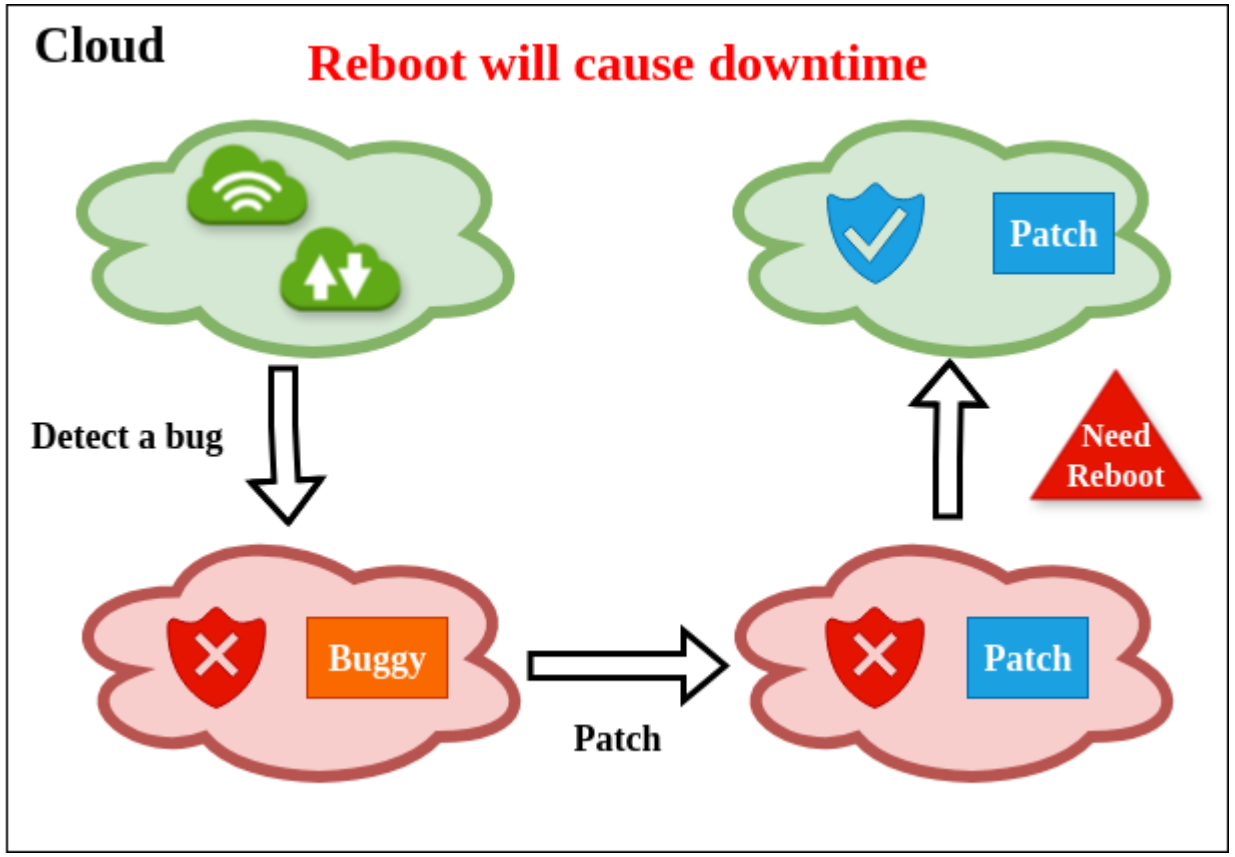
FortifyPatch: Towards Tamper-Resistant Live Patching in Linux-Based Hypervisor

Zhenyu Ye [1], Lei Zhou [2], Fengwei Zhang [3], Wenqiang Jin [1],
Zhenyu Ning [1], Yupeng Hu [1], and Zheng Qin [1]

Hunan University [1]
National University of Defense [2]
Southern University of Science and Technology [3]

Introduction





Average cost of downtime : $5600/min [2].

[1] https://www.cvedetails.com/product/47/Linux-Linux-Kernel.html?vendor_id=33
[2] Lei Zhou, et al. 2020. KShot: Live kernel patching with SMM and SGX. In Proceedings of the 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'20).

Live Patching

Patching the Linux kernel at runtime.

- Rely on the kernel.
  - might be compromised.

- KShot[1]: A kernel hot patching mechanism based on x86 SMM and Intel SGX.
  - SMM is not a generic mode.
  - fails to protect the patch afterwards.

[1] Lei Zhou, et al. 2020. KShot: Live kernel patching with SMM and SGX. In Proceedings of the 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'20).

Live Patching

Can we protect patch from tamper in runtime?

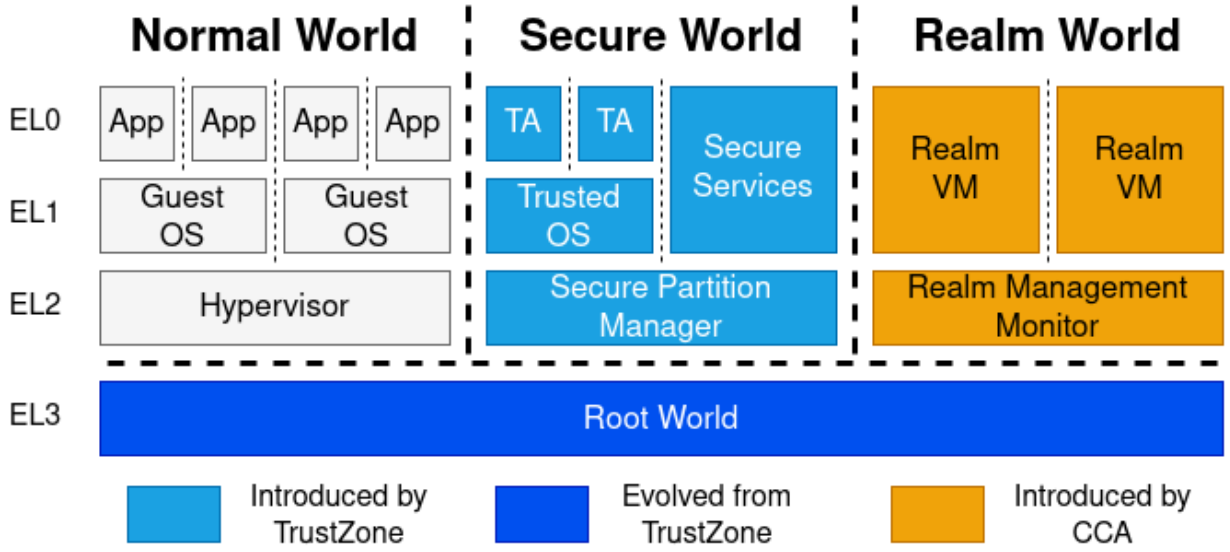
- Using a memory access control mechanism.
  - ✓ Arm TrustZone Address Space Controller (TZC).
  - ✓ RISC-V Physical Memory Protection (PMP).

Has limited region number

Lack flexibility!

Confidential Compute Architecture (CCA)

Protect virtual machines under untrusted hypervisors.



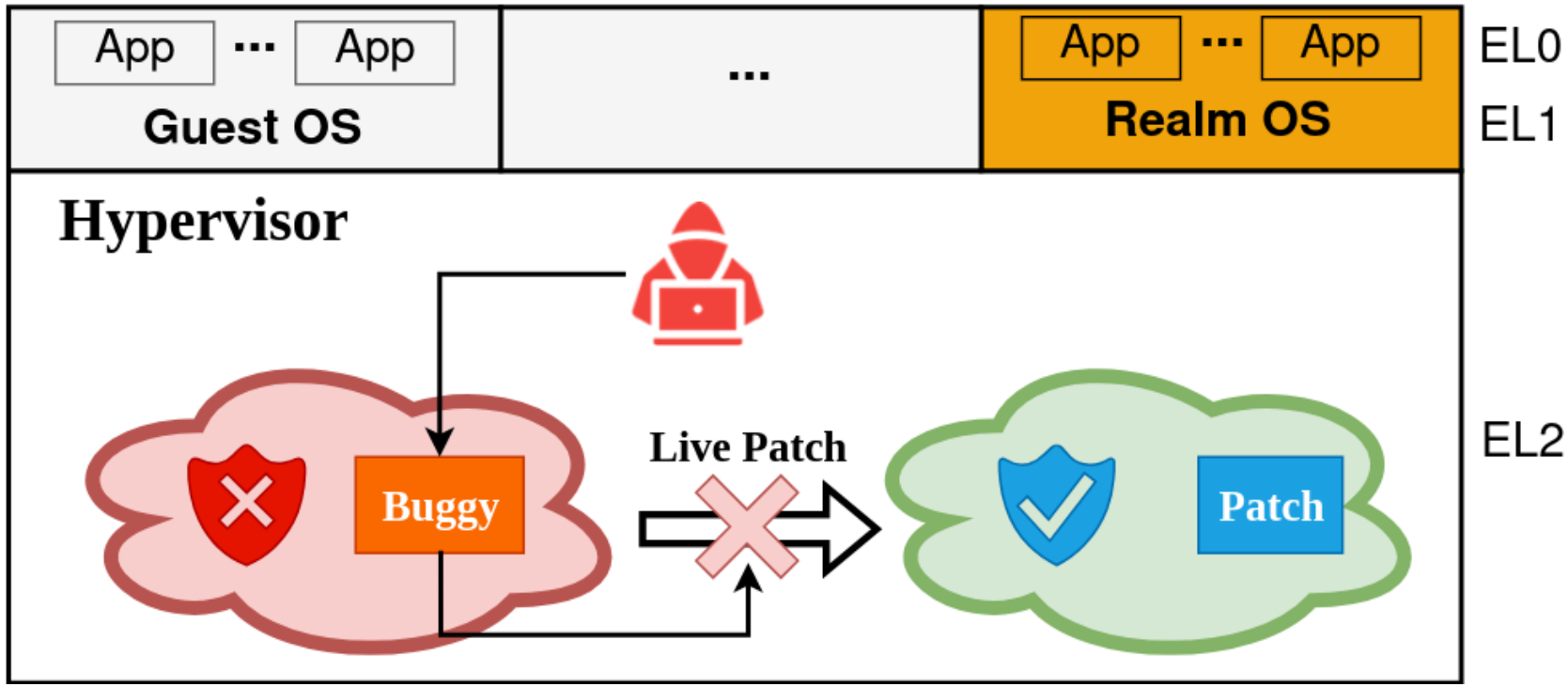
Arm Confidential Computing Architecture

- Granule Protection Table (GPT): Protect memory flexibly. (4 KB)
- Granule Protection Check (GPC): Validates privileged access.
- Granule Protection Fault (GPF): Blocks illegal access.

Confidential Compute Architecture (CCA)

Challenges

C1: The attacker and the patch are sitting in the same privilege.

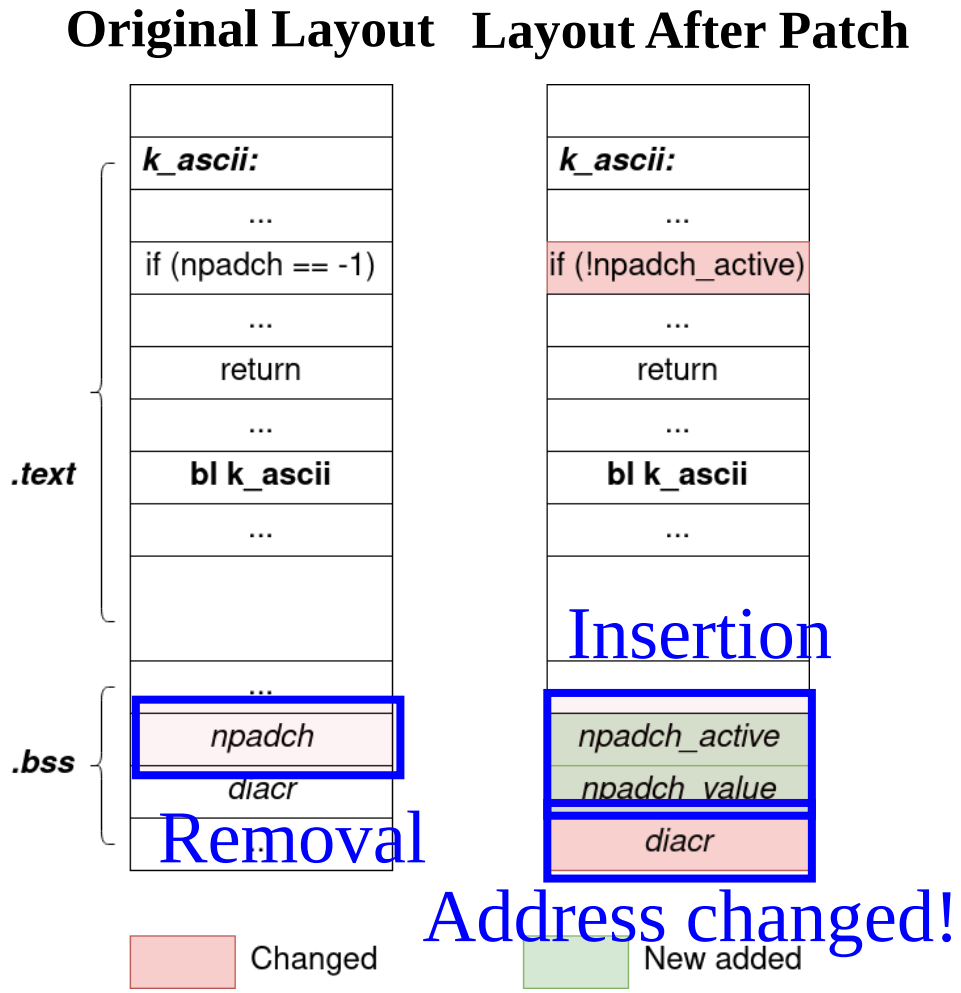


Challenges

C2: Specific patches may cause changes in the memory layout.

```
--- a/drivers/tty/vt/keyboard.c
+++ b/drivers/tty/vt/keyboard.c
- static int npadch = -1;

+ static bool npadch_active;
+ static unsigned int npadch_value;

static unsigned int diacr;

@@ -852,10 +856,12 @@ static void k_ascii(struct vc_data *vc,
      unsigned char value, char up_flag)
     base = 16;
 }

- if (npadch == -1)
-     npadch = value;
- else
-     npadch = npadch * base + value;
+ if (!npadch_active) {
+     npadch_value = 0;
+     npadch_active = true;
+ }
```

A subset of patch for CVE-2020-13974

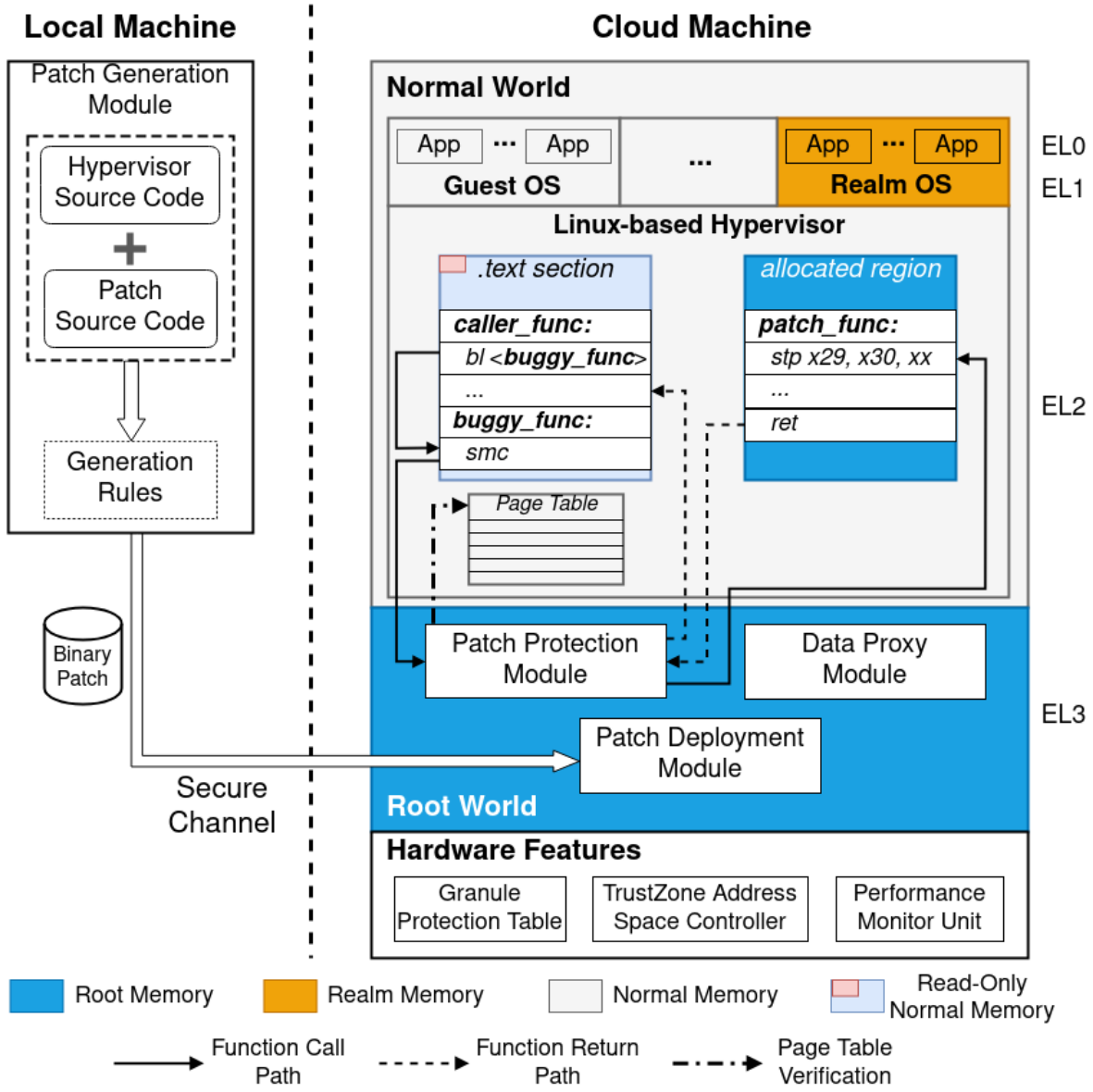


Challenges

C3: Adopting the security policies introduces notable performance overhead.

- Executing a patch requires modifying the GPT.
- Accessing data from the patch causes frequent traps.
- Switching worlds requires a context switch.

Overview of FortifyPatch



- Patch Generation Module
  - ✓ Generates a binary patch.
- Patch Deployment Module
  - ✓ Receives and applies the patch.
- Patch Protection Module
  - ✓ Protects the patch.
- Data Proxy Module
  - ✓ Facilitates data access.

Design

C1: The attacker and the patch are sitting in the same privilege.

S1: Make the patch executable but **NOT** writable to kernel.

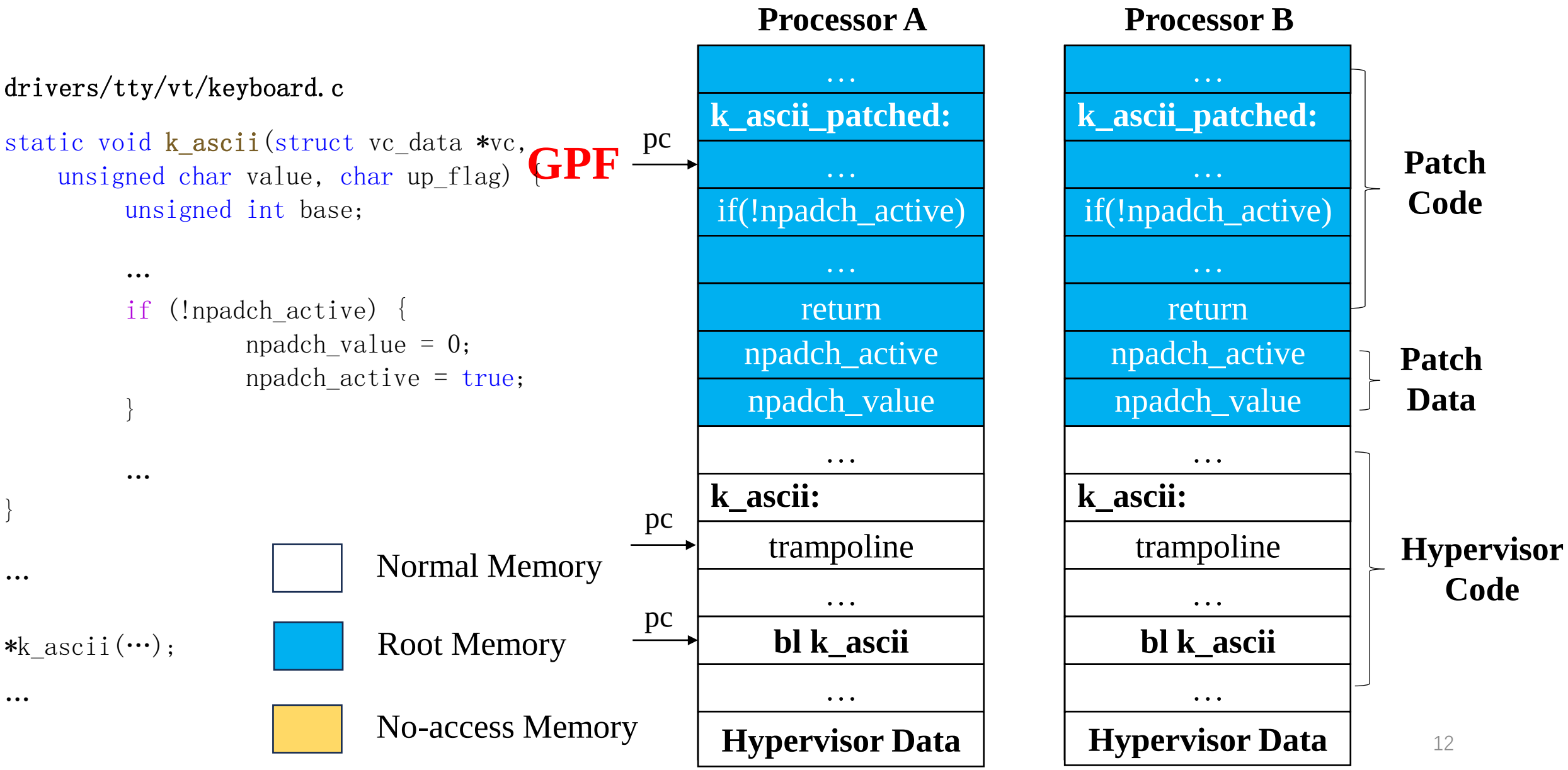
- Multi-GPT scheme.

Patch Protection Module

drivers/tty/vt/keyboard.c

```
static void k_ascii(struct vc_data *vc,
    unsigned char value, char up_flag) {
        unsigned int base;

        ...
        if (!npadch_active) {
                npadch_value = 0;
                npadch_active = true;
        }

        ...
}

...

*k_ascii(…);

...
```

GPF



Patch Protection Module

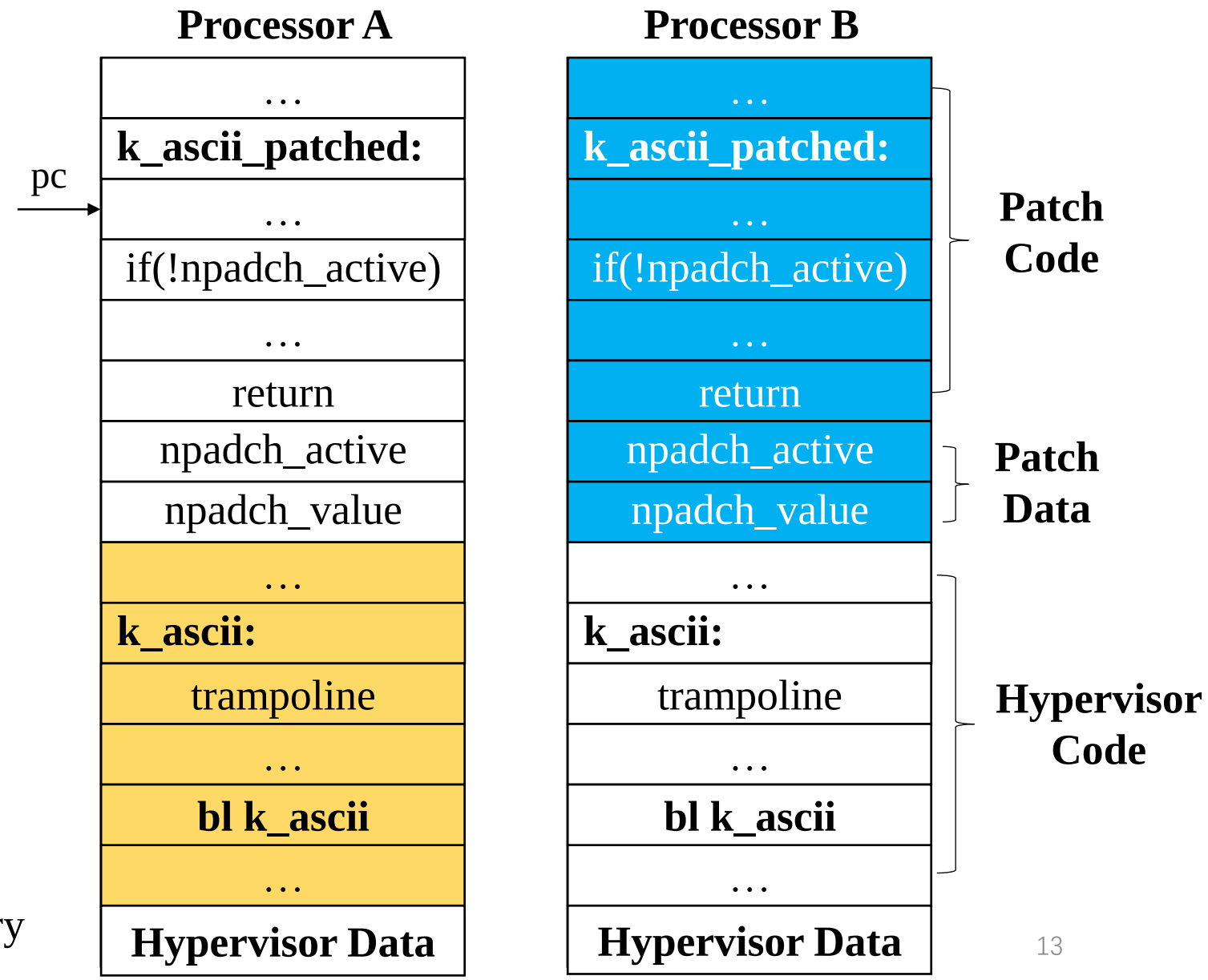
drivers/tty/vt/keyboard.c

```
static void k_ascii(struct vc_data *vc,
    unsigned char value, char up_flag) {
        unsigned int base;

        ...
        if (!npadch_active) {
                npadch_value = 0;
                npadch_active = true;
        }

        ...
}

...

*k_ascii(…);

...
```

Normal Memory

Root Memory

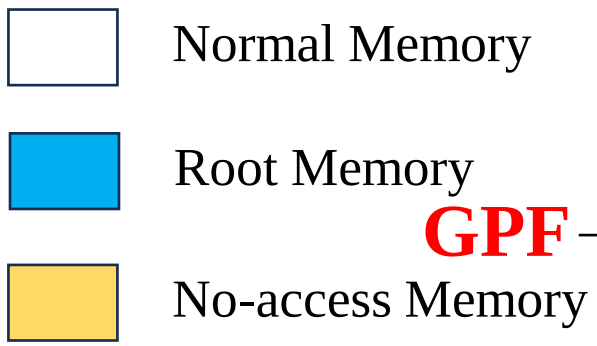
No-access Memory

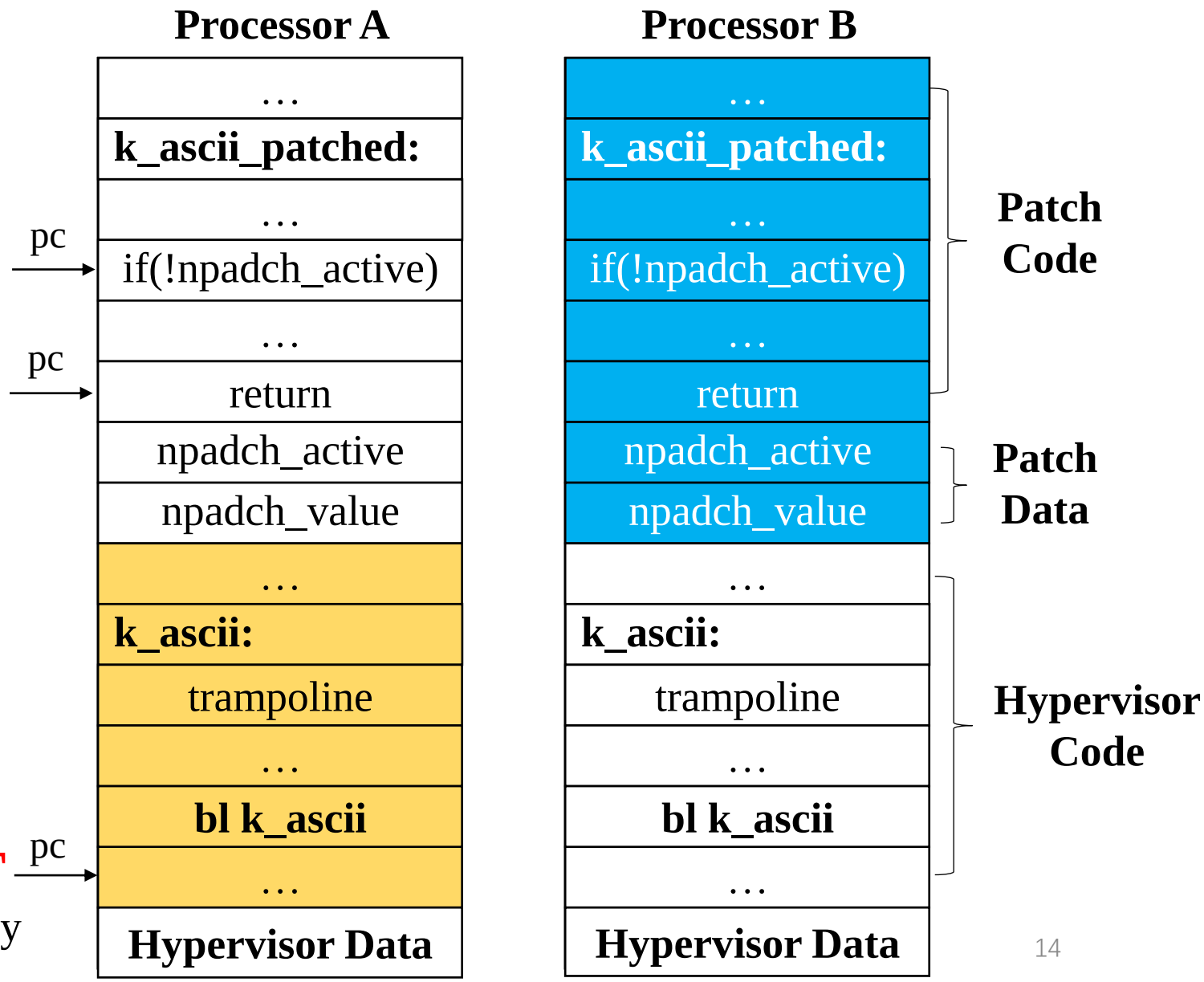


Patch Protection Module

drivers/tty/vt/keyboard.c

```
static void k_ascii(struct vc_data *vc,
    unsigned char value, char up_flag) {
        unsigned int base;

        ...
        if (!npadch_active) {
                npadch_value = 0;
                npadch_active = true;
        }

        ...
}

...

*k_ascii(…);

...
```





Patch Protection Module

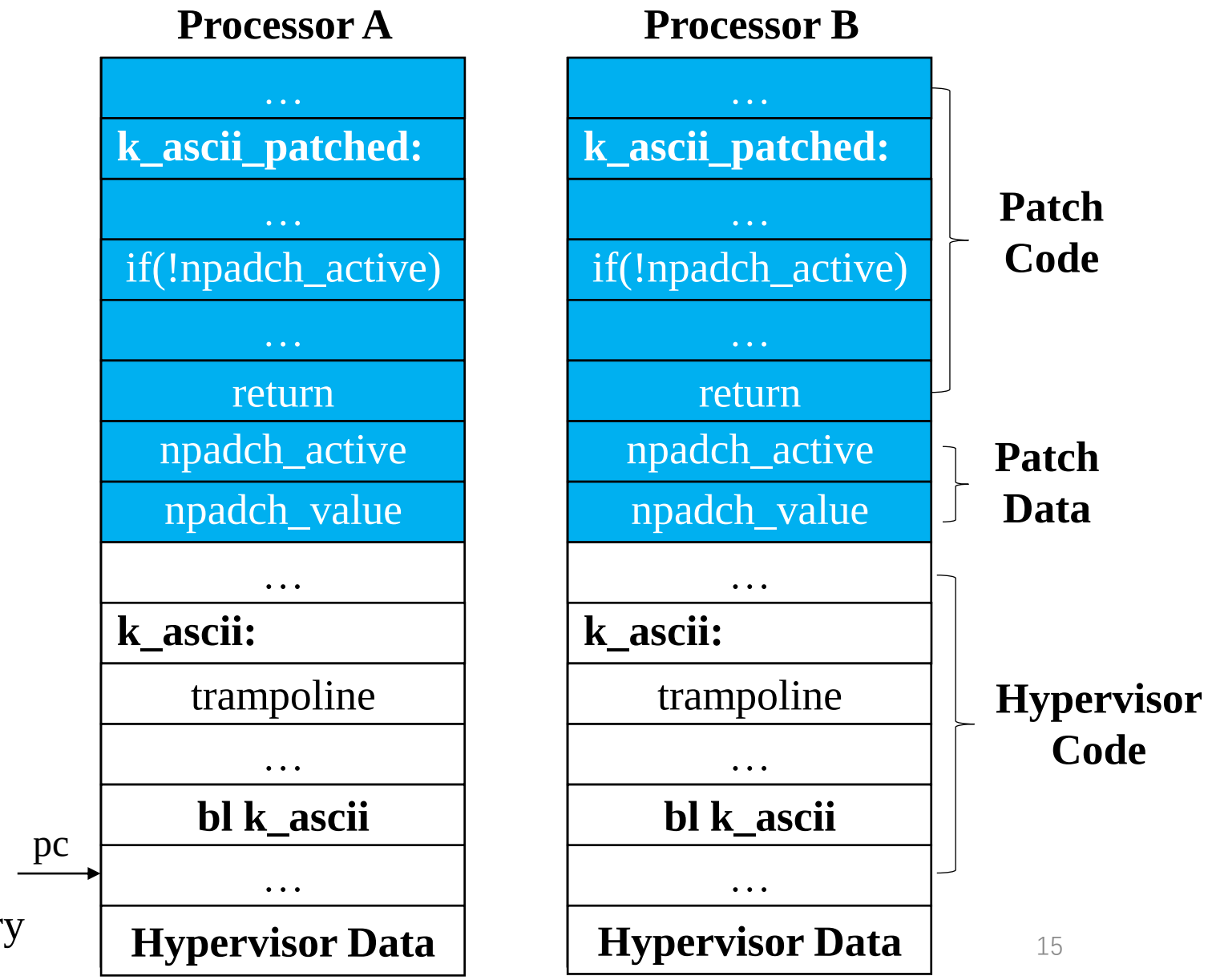
drivers/tty/vt/keyboard.c

```
static void k_ascii(struct vc_data *vc,
    unsigned char value, char up_flag) {
        unsigned int base;

        ...
        if (!npadch_active) {
                npadch_value = 0;
                npadch_active = true;
        }

        ...
}

...

*k_ascii(…);

...
```

Normal Memory

Root Memory

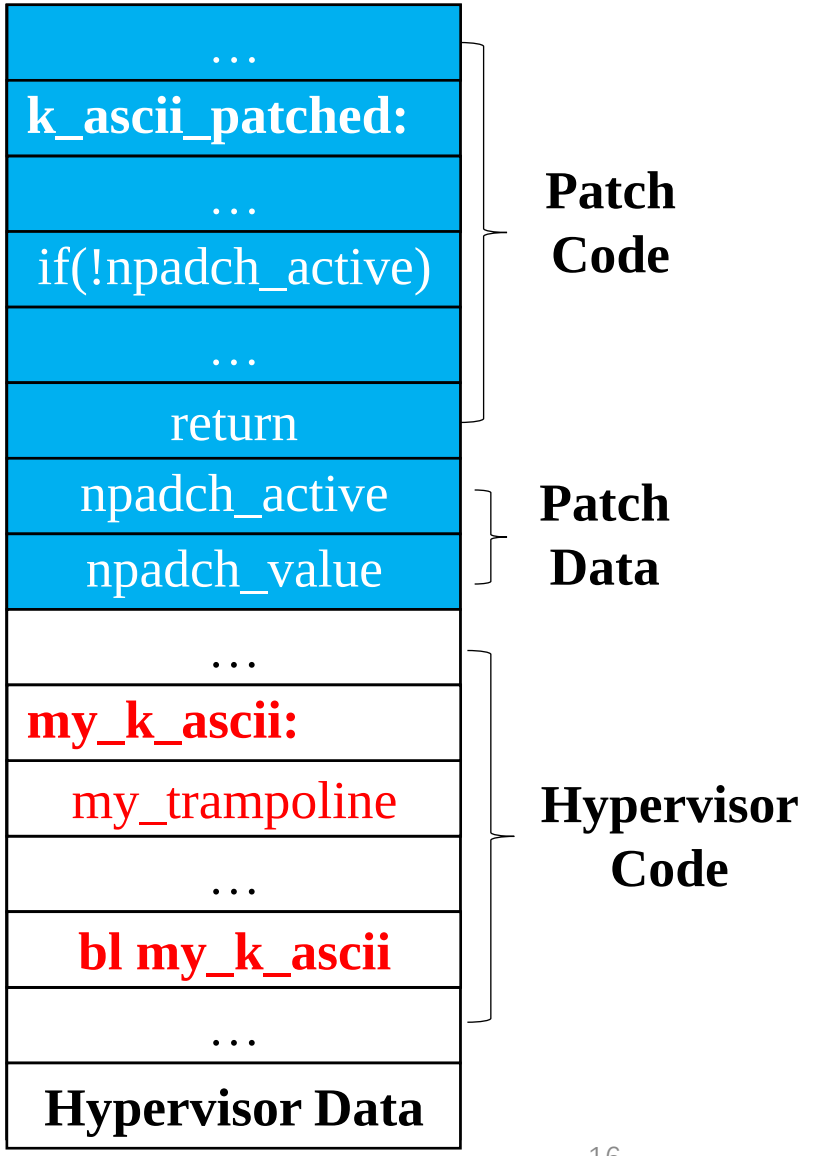
No-access Memory



Patch Protection Module

What if the patch is bypassed ?

- Modify hypervisor code.
  - ✓ The kernel .*text* section uses continuous physical memory.
  - ✓ Mark the .*text* section as read-only. (**TZC**)
- Modify page table.
  - ✓ Verify the related page table entries for the .*text* section. (**PMU** overflow interrupt)

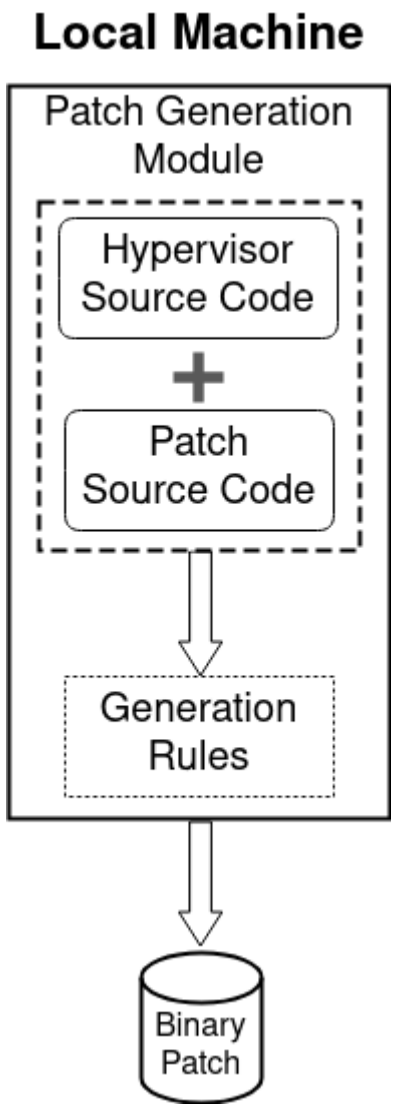


Design

C2: Specific patches may cause changes in the memory layout.

S2: Adopt different mechanisms for various scenarios.

Patch Generation Module

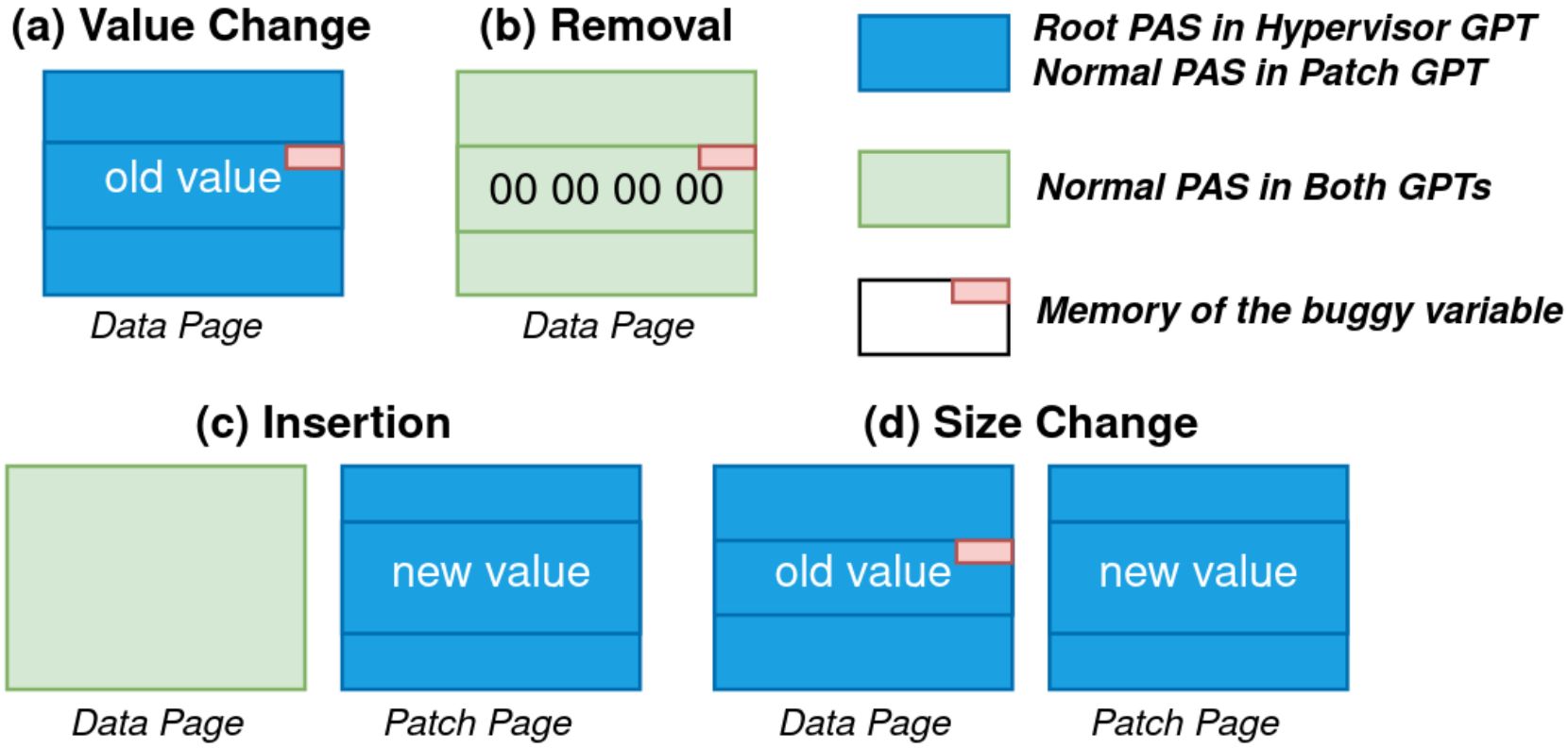


Patches for Global Variables:

Data modification type	Handler
Data Value Change	Record the data address and the new value in the patch.
Data Size Change	Place the changed data in a separate location and trap old data access to the new location.
Data Insertion	Place the changed data in a separate location and update reference to new data.
Data Removal	Zeroing the corresponding memory.

Patch Protection Module

GPT safeguards patches for global variables.



Design

C3: Adopting the security policies introduces notable performance overhead.

- Reducing the number of traps required.
- Reducing the operations in the GPT switching.
- Reducing the overhead of context save/restore.

For a C program with getpid(), these policies result in reduction of 66% GPT switches and decrease of over 70% overhead for each switch.

Evaluation

RQ1: The TCB Introduced by FortifyPatch.

Component	Language	Location	LoC
Patch Generation Module	Python	Local	661
Patch Deployment Module	C、Assembly	Hypervisor	312
Patch Protection Module	C、Assembly	Hypervisor	350
Data Proxy Module	C、Assembly	Hypervisor	696
Total TCB			**1358**

The code size of TF-A : about 310K LoC.

Evaluation

RQ2: Effectiveness.

2013 – 2023: 2000 submitted CVE with 1385 identified patches.

Number	Note	Success
122	Not for Arm64	N/A
862	Devices drivers (not support in FVP)	N/A
62	Compile-time expanding semantics	✗
9	Makefile and Kconfig files	✗
3	Init functions	✗
327	Success	✓

Can deploy approximately 81.5% of the CVE patches.

Evaluation

RQ3: Performance.

No real-world device with Arm CCA.

Raspberry PI 3B+

Benchmark prototype.

- *sysbench*.
- *UnixBench*.
- *lmbench*.

3 real-world app.

- *Memcached*.
- *Nginx*.
- *Apache*.

Performance.

- Stage-2 translation replace GPC.
- TTBR_EL2 replace GPTBR_EL3

10 real-world CVE patches.

CVE	Patch Function
CVE-2014-0196	n_tty_write
CVE-2016-0728	join_session_keyring
CVE-2016-7916	environ_read
CVE-2017-17052	mm_init
CVE-2018-1095	ext4_xattr_check_entries
CVE-2018-10087	kernel_wait4
CVE-2018-13405	inode_init_owner
CVE-2019-9213	expand_downwards
CVE-2020-13974	k_shift/k_ascii
CVE-2022-2978	Inode_init_always

Evaluation

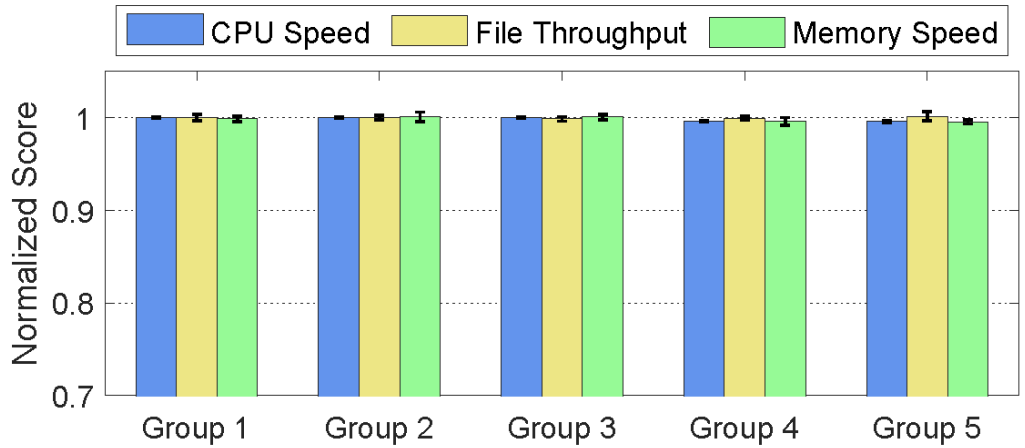
RQ3: Performance.

(a) With a Single Patch.

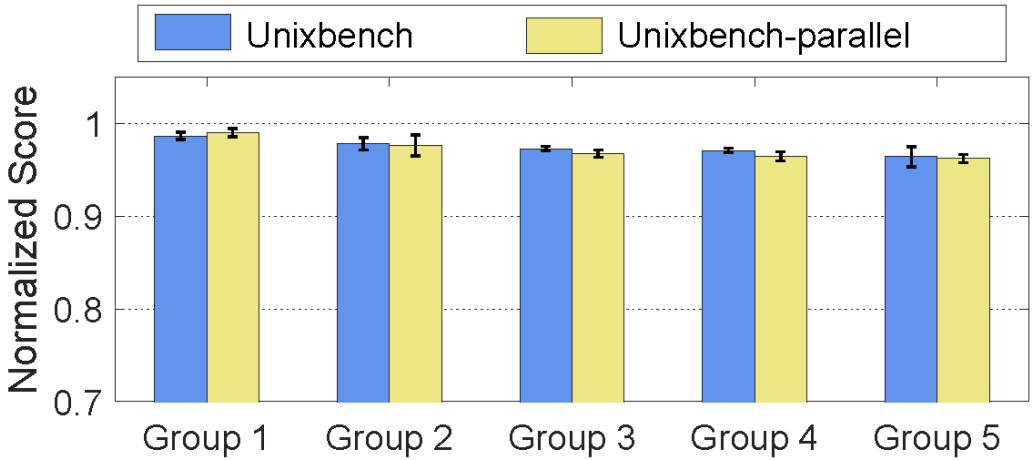
Index	CVE	sysbench	Unixbench	lmbench
1	CVE-2014-0196	1.50×10^2	5.70×10^2	7.20×10^2
2	CVE-2016-0728	0	18	2
3	CVE-2016-7916	0	2	0
4	CVE-2017-17052	1.03×10^3	4.91×10^7	1.03×10^5
5	CVE-2018-1095	98	2	0
6	CVE-2018-10087	1.62×10^3	3.97×10^7	1.28×10^5
7	CVE-2018-13405	6.87×10^2	6.68×10^6	3.48×10^6
8	CVE-2019-9213	8.28×10^2	2.23×10^7	2.97×10^4
9	CVE-2020-13974	2.41×10^5	2.52×10^6	6.07×10^5
10	CVE-2022-2978	1.27×10^4	1.17×10^7	3.02×10^6

(b) With Multiple Patches.

Group	Patch Indices	sysbench	Unixbench	lmbench
1	2, 8	8.46×10^2	2.23×10^7	2.97×10^4
2	2, 4, 8, 10	1.38×10^4	8.16×10^7	3.30×10^6
3	1, 2, 4, 7, 8, 10	1.61×10^4	8.68×10^7	5.23×10^6
4	1, 2, 3, 4, 7, 8, 9, 10	2.58×10^5	8.93×10^7	5.79×10^6
5	all	2.59×10^5	1.18×10^8	5.99×10^6



sysbench



UnixBench

Evaluation

RQ3: Performance.

(a) With a Single Patch.

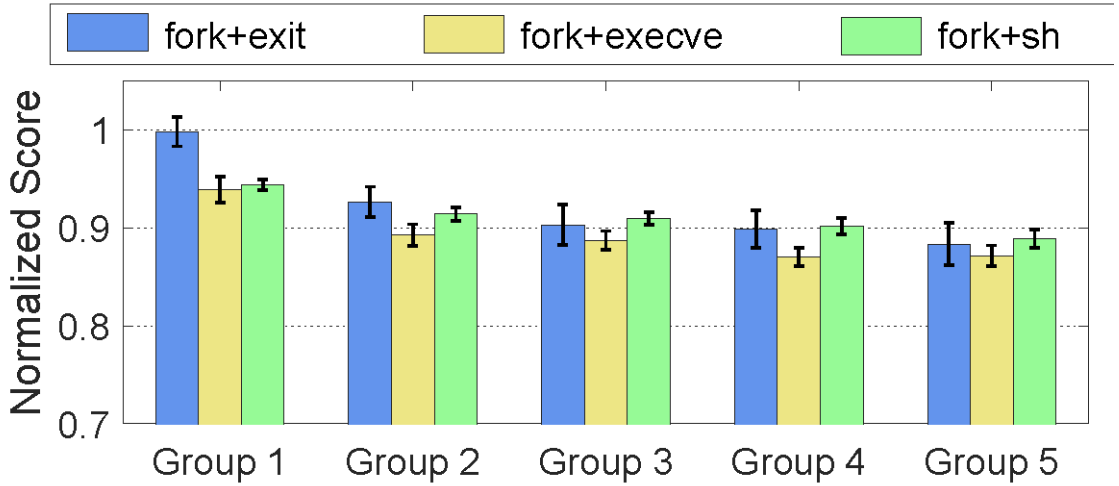
Index	CVE	sysbench	Unixbench	lmbench
1	CVE-2014-0196	1.50×10^2	5.70×10^2	7.20×10^2
2	CVE-2016-0728	0	18	2
3	CVE-2016-7916	0	2	0
4	CVE-2017-17052	1.03×10^3	4.91×10^7	1.03×10^5
5	CVE-2018-1095	98	2	0
6	CVE-2018-10087	1.62×10^3	3.97×10^7	1.28×10^5
7	CVE-2018-13405	6.87×10^3	6.68×10^6	3.48×10^6
8	CVE-2019-9213	8.28×10^2	2.23×10^7	2.97×10^4
9	CVE-2020-13974	2.41×10^5	2.52×10^6	6.07×10^5
10	CVE-2022-2078	1.27×10^4	1.17×10^7	3.02×10^6

Worst case: 12.9%

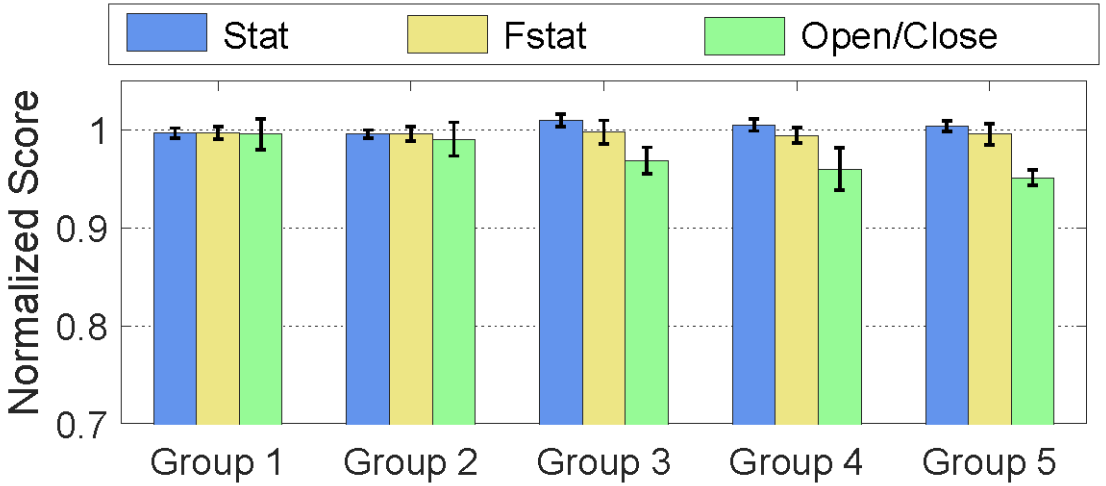
Average overhead: 0.98%

(b) With Multiple Patches.

Group	Patch Indices	sysbench	Unixbench	lmbench
1	2, 8	8.46×10^2	2.23×10^7	2.97×10^4
2	2, 4, 8, 10	1.38×10^4	8.16×10^7	3.30×10^6
3	1, 2, 4, 7, 8, 10	1.61×10^4	8.68×10^7	5.23×10^6
4	1, 2, 3, 4, 7, 8, 9, 10	2.58×10^5	8.93×10^7	5.79×10^6
5	all	2.59×10^5	1.18×10^8	5.99×10^6



lmbench

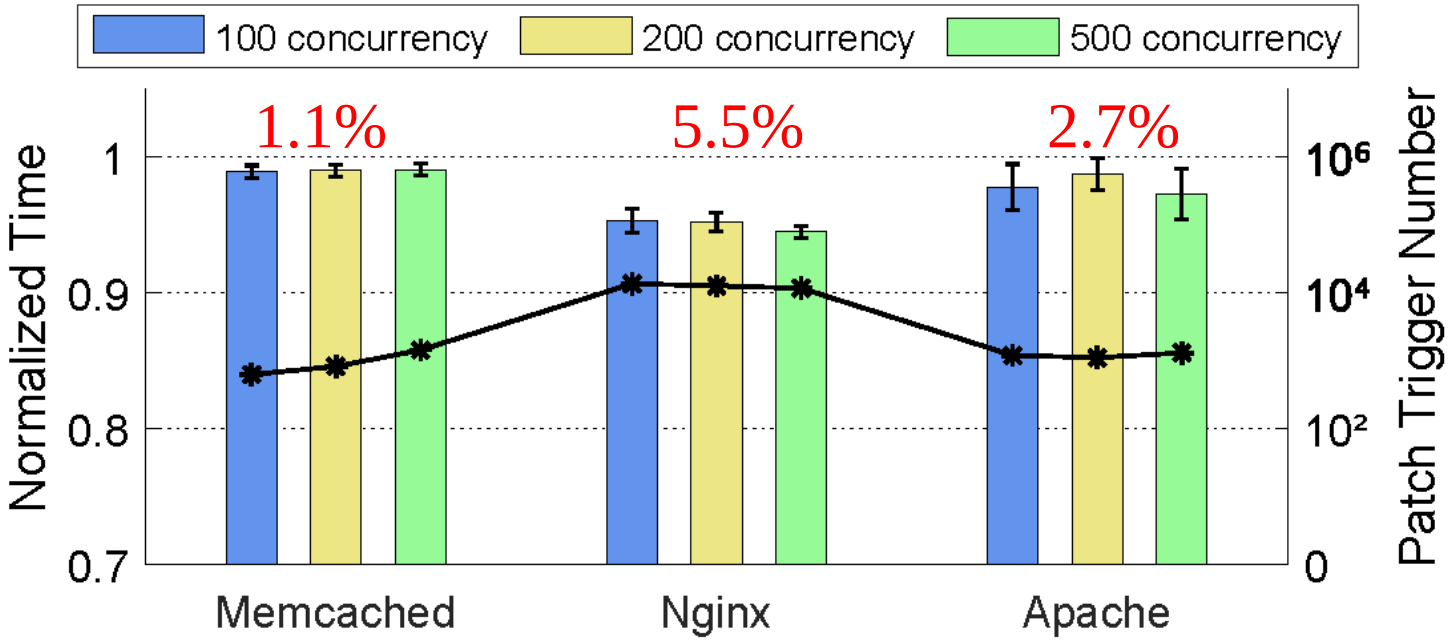


lmbench

Evaluation

RQ3: Performance.

- All 10 patch deployed.
- Run with various concurrency levels.



Evaluation

RQ4: Compatibility.

Industry-level confidential computing prototype : **Samsung Islet**.

- Integrate FortifyPatch into firmware.
- Deploy all the 10 CVE patches.
- Launch a realm VM.
- Run provided sdk-example in realms.

Evaluation

RQ5: Comparing with others.

System Name	Patch Level	Secure Patching	Tamper Resistance	Global Data	Downtime	Memory	Overhead
KUP[1]	kernel	✘	✘	✘	2.4 s/kernel	56 GB	／
RapidPatch[2]	instruction	✘	✘	✘	7.5 μs/23LoC	18 KB	2.2%~9.1%
kpatch[3]	function	✘	✘	✘	45.6 ms/patch	20 MB	／
Kshot[4]	function	✓	✘	✘	50 μs/patch	18 MB	3%
FortifyPatch	function	✓	✓	✓	166.92 μs/patch	16 MB	0.1%~6.4%

Patch Size : about 1 KB.

[1] Sanidhya Kashyap, et al. 2016. Instant OS updates via userspace checkpoint-and-restart. In Proceedings of the 2016 USENIX Annual Technical Conference (ATC'16).
[2] Yi He, et al. 2022. RapidPatch: Firmware hotpatching for real-time embedded devices. In Proceedings of the 31st USENIX Security Symposium (Security'22).
[3] Red Hat. 2023. kpatch: dynamic kernel patching. https://github.com/dynup/kpatch.
[4] Lei Zhou, et al. 2020. KShot: Live kernel patching with SMM and SGX. In Proceedings of the 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'20).

Conclusion

- Present FortifyPatch, a tamper-resistant live patching system designed to persistently patch Linux-based hypervisors at runtime.
- FortifyPatch utilizes well-designed traps to reduce the impact on the number of affected instructions for the practical issue related to patching global variables in live patching.
- FortifyPatch protects patches with 0.98% and 3.1% overhead on average across indicative benchmarks and real-world applications, respectively.

Thanks for listening!

Q&A!

Contact : yezhenyu@hnu.edu.cn

Artifact : https://doi.org/10.5281/zenodo.126572

Data Proxy Module

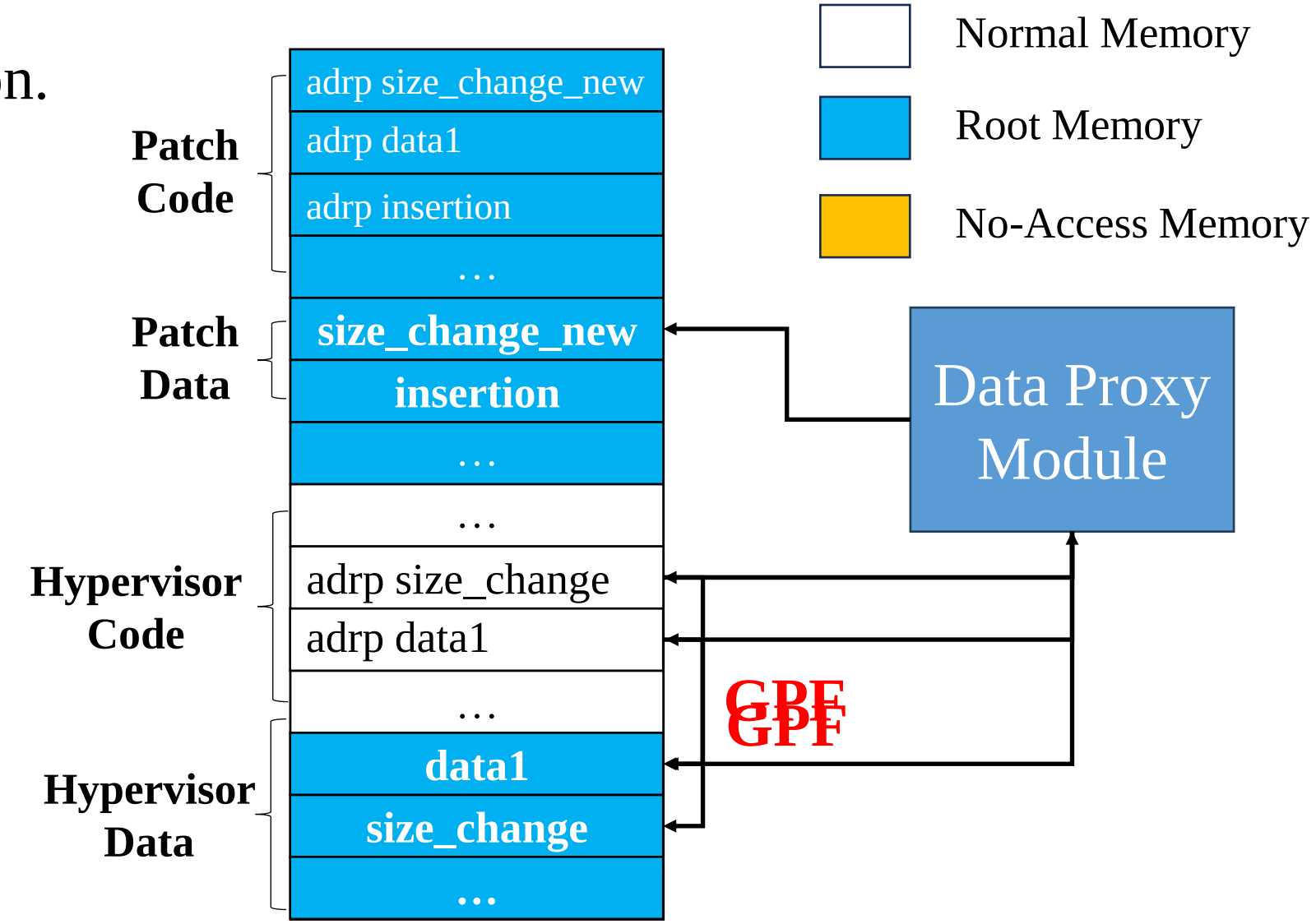
Data located in new location.

Insertion.

- Only accessed in patch code.

Size change.

- 4KB of origin location set to root.



Data Proxy Module

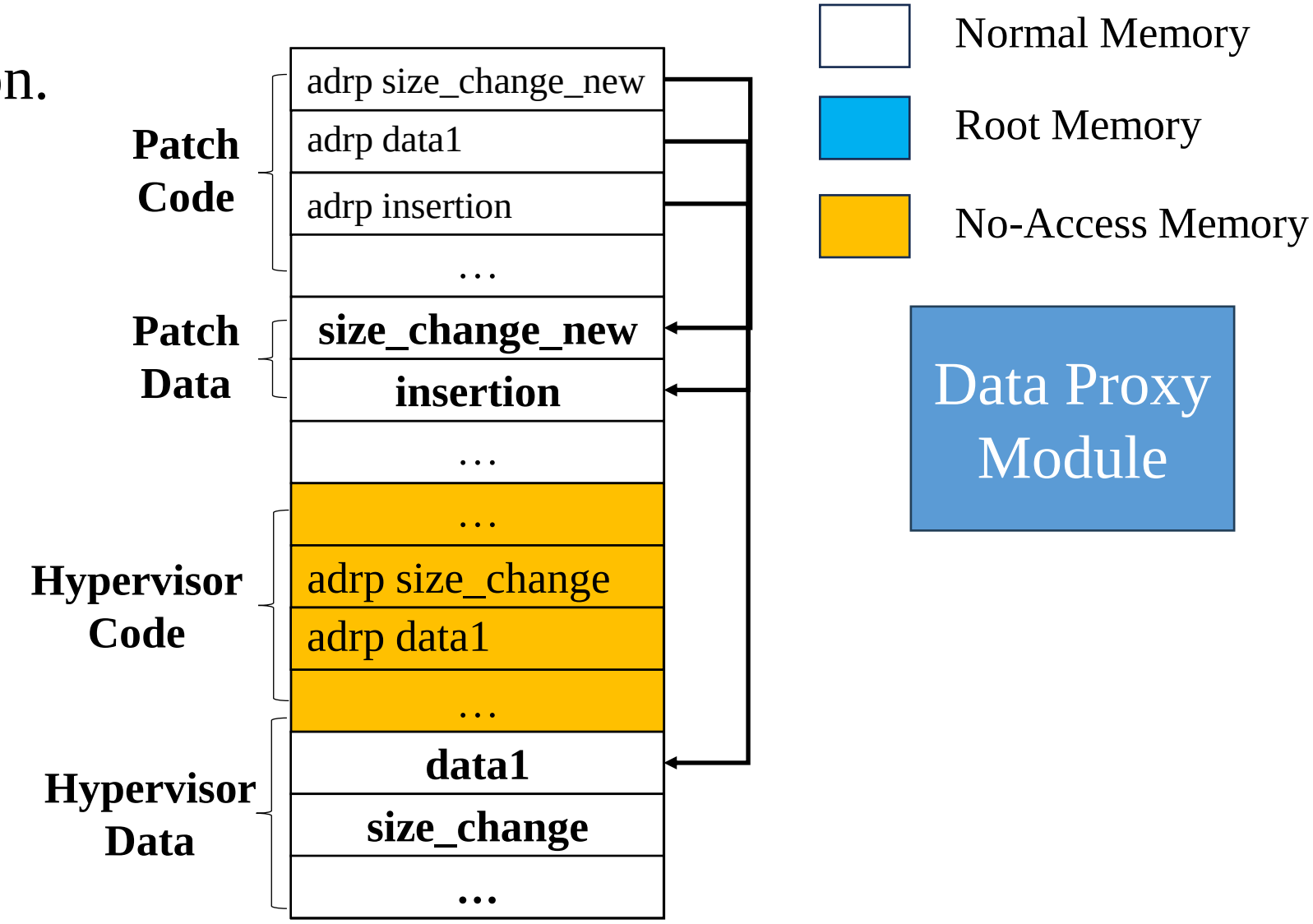
Data located in new location.

Insertion.

- Only accessed in patch code.

Size change.

- 4KB of origin location set to root.



Data Proxy Module

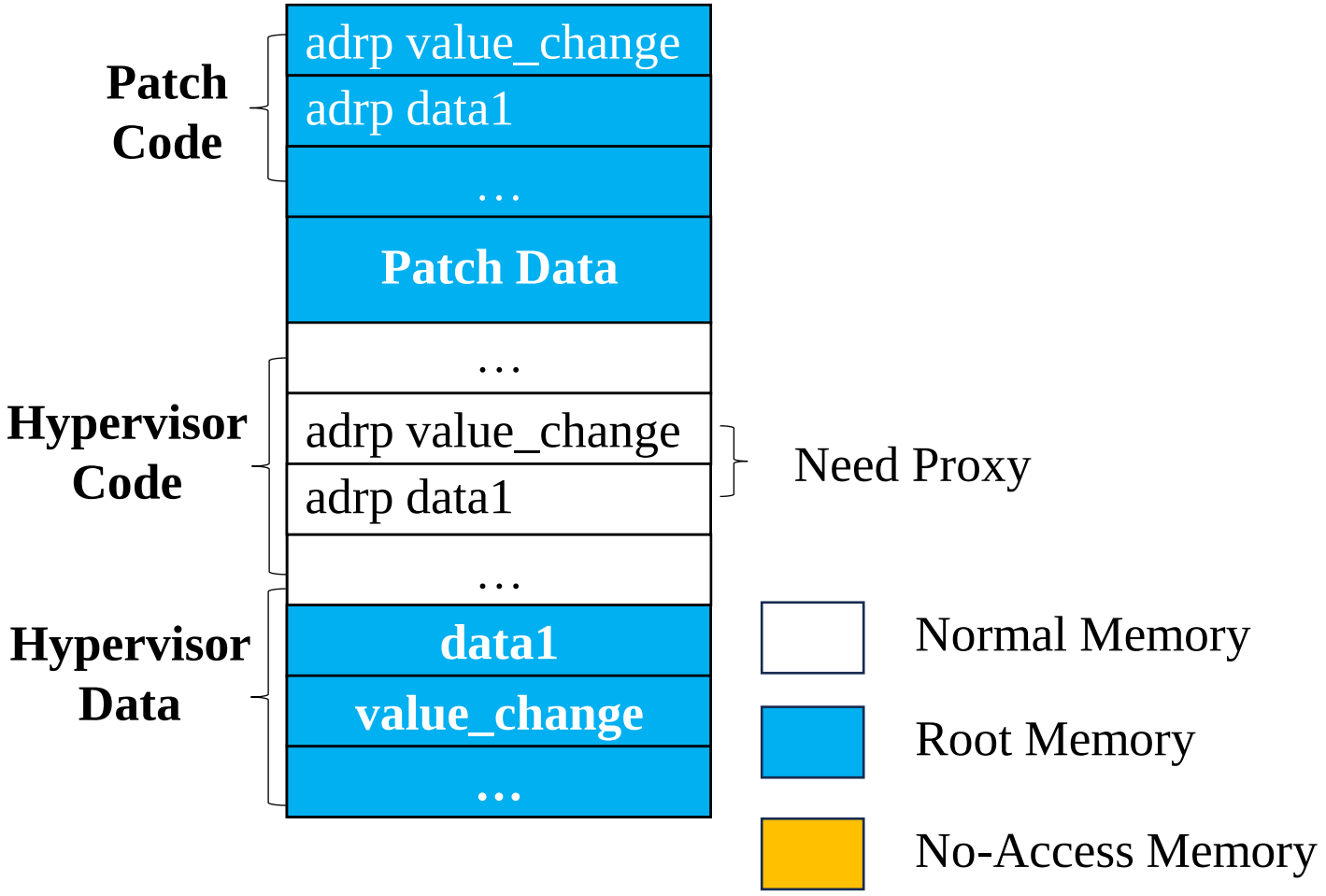
Data located in old location.

Removal.

- No instruction will access it.

Value change.

- 4KB of origin location set to root.



Data Proxy Module

Data located in old location.

Removal.

- No instruction will access it.

Value change.

- 4KB of origin location set to root.