TECHNISCHE UNIVERSITÄT DRESDEN

Department of Computer Science Institute of System Architecture, Operating Systems Group

SECURITY ARCHITECTURES

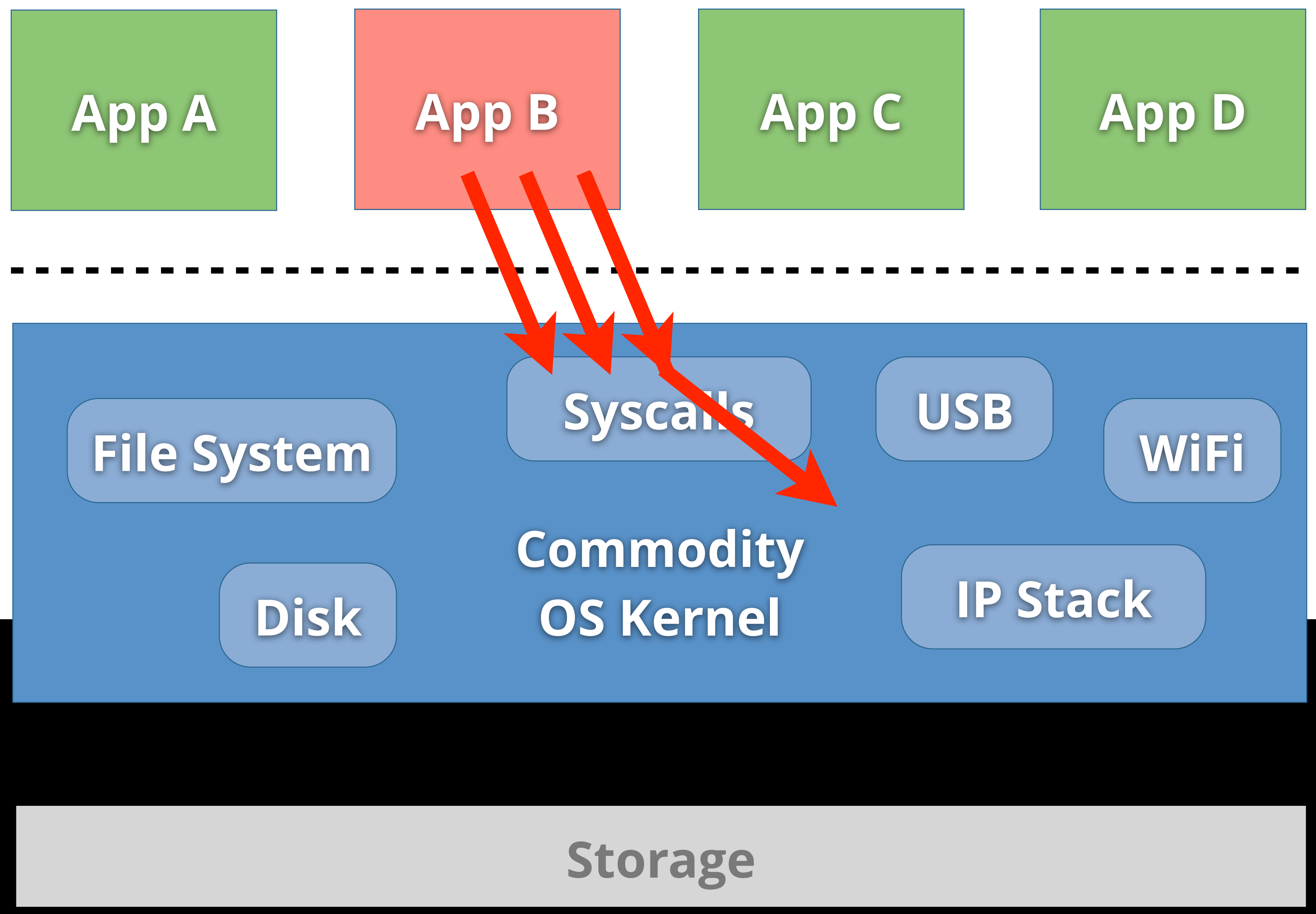
CARSTEN WEINHOLD

CLASSICAL ARCHITECTURES

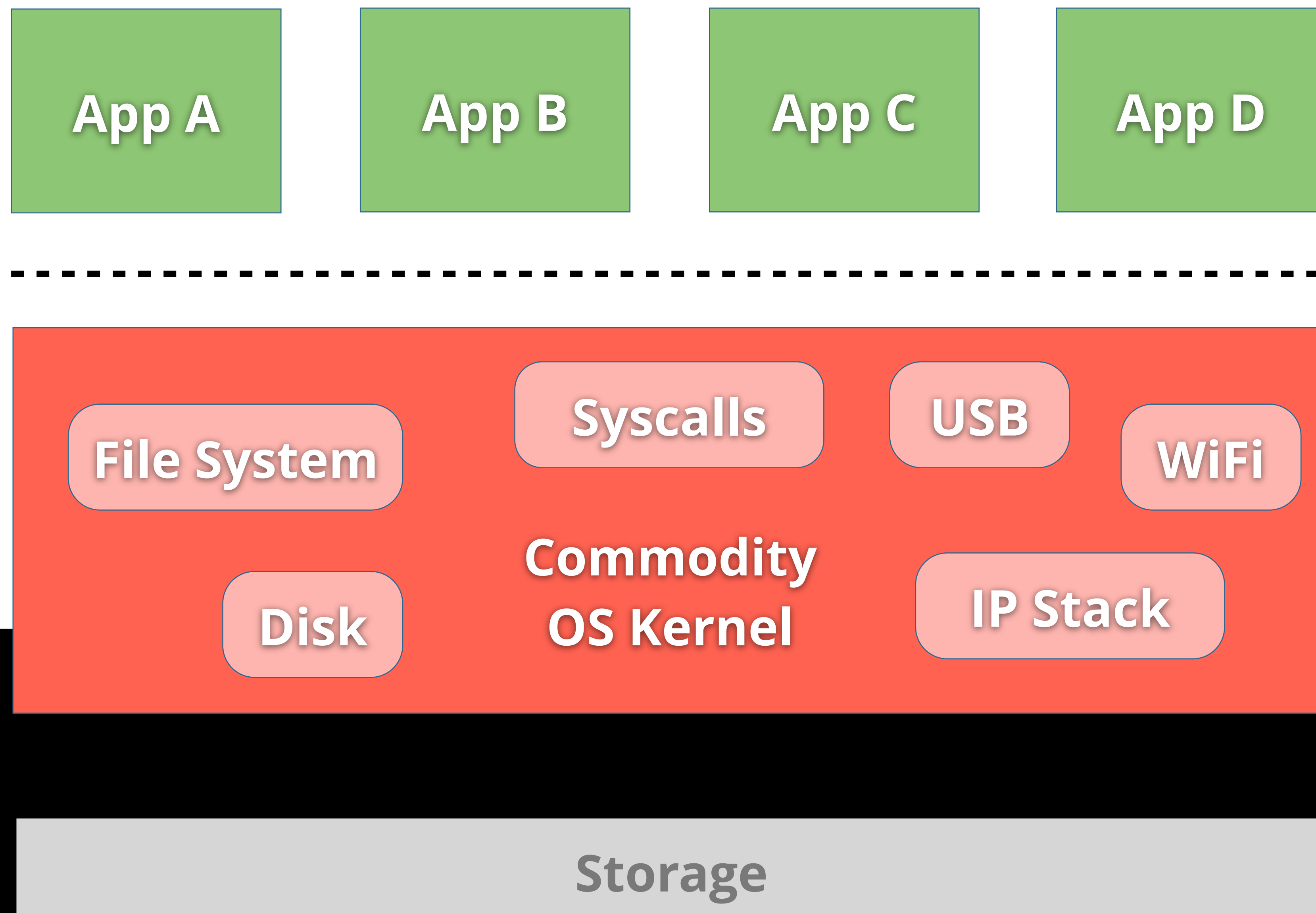
ISOLATION

- Isolation in commodity OSes for PCs:
  - Based on user accounts
  - Same privileges for all apps
  - No isolation within applications
  - Permissive interfaces (e.g., ptrace to manipulate other address spaces)

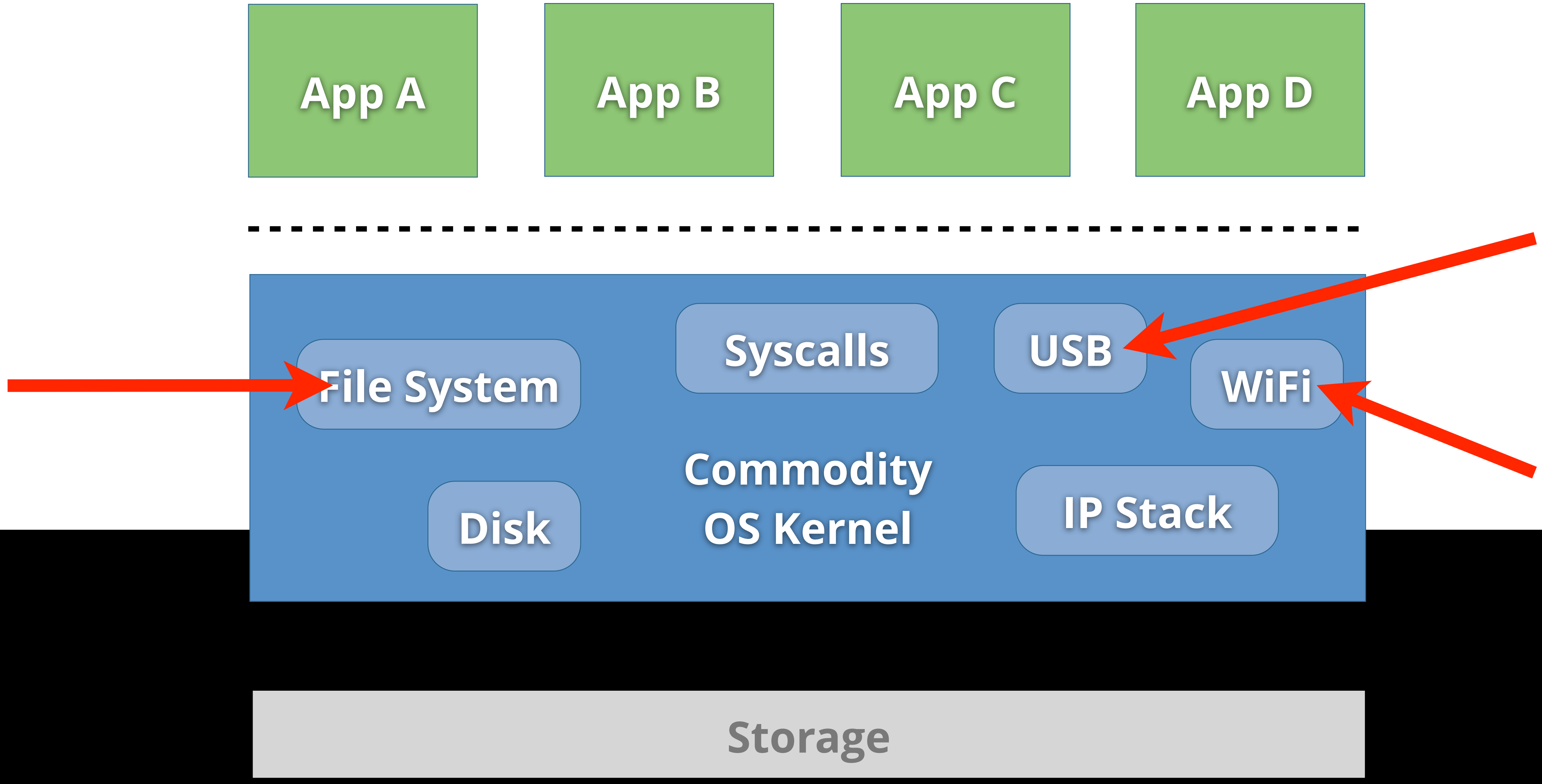
KERNEL ATTACK VECTOR



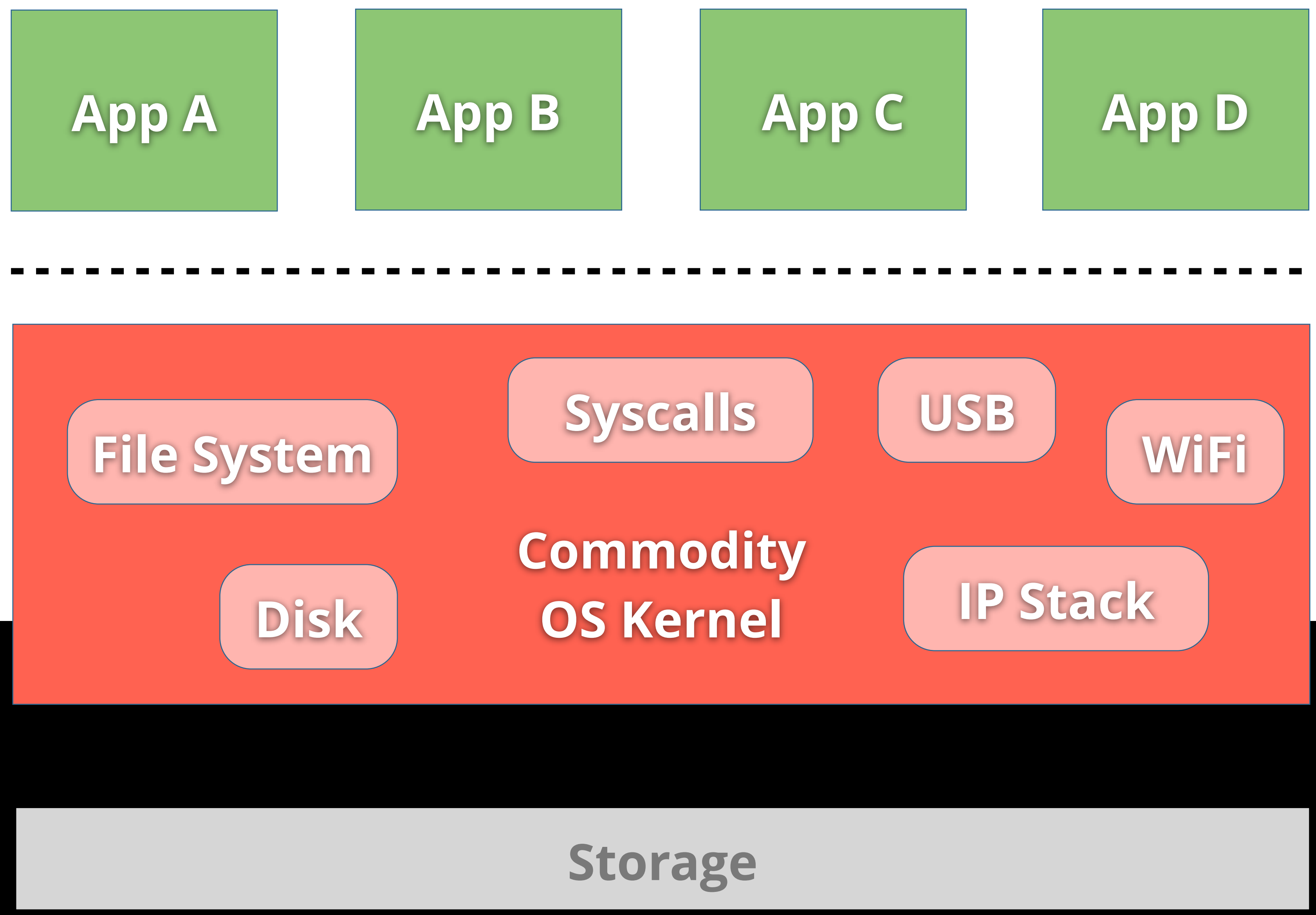
KERNEL ATTACK VECTOR



KERNEL ATTACK VECTOR



KERNEL ATTACK VECTOR



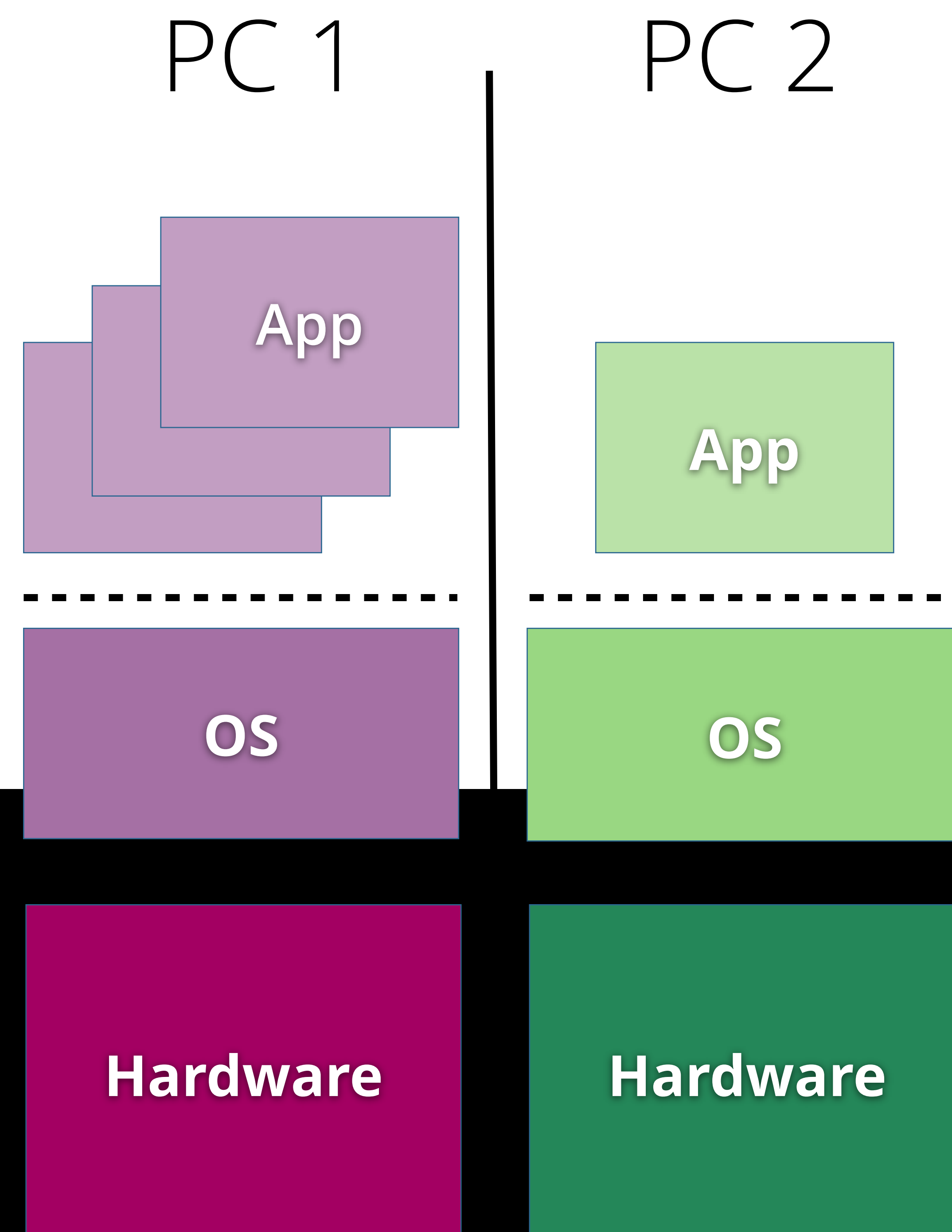
TECHNISCHE UNIVERSITÄT DRESDEN

ISOLATION

- Isolation in commodity OSes for PCs:
  - Based on user accounts
  - Same privileges for all apps
  - No isolation within applications
  - Permissive interfaces (e.g., ptrace to manipulate other address spaces)

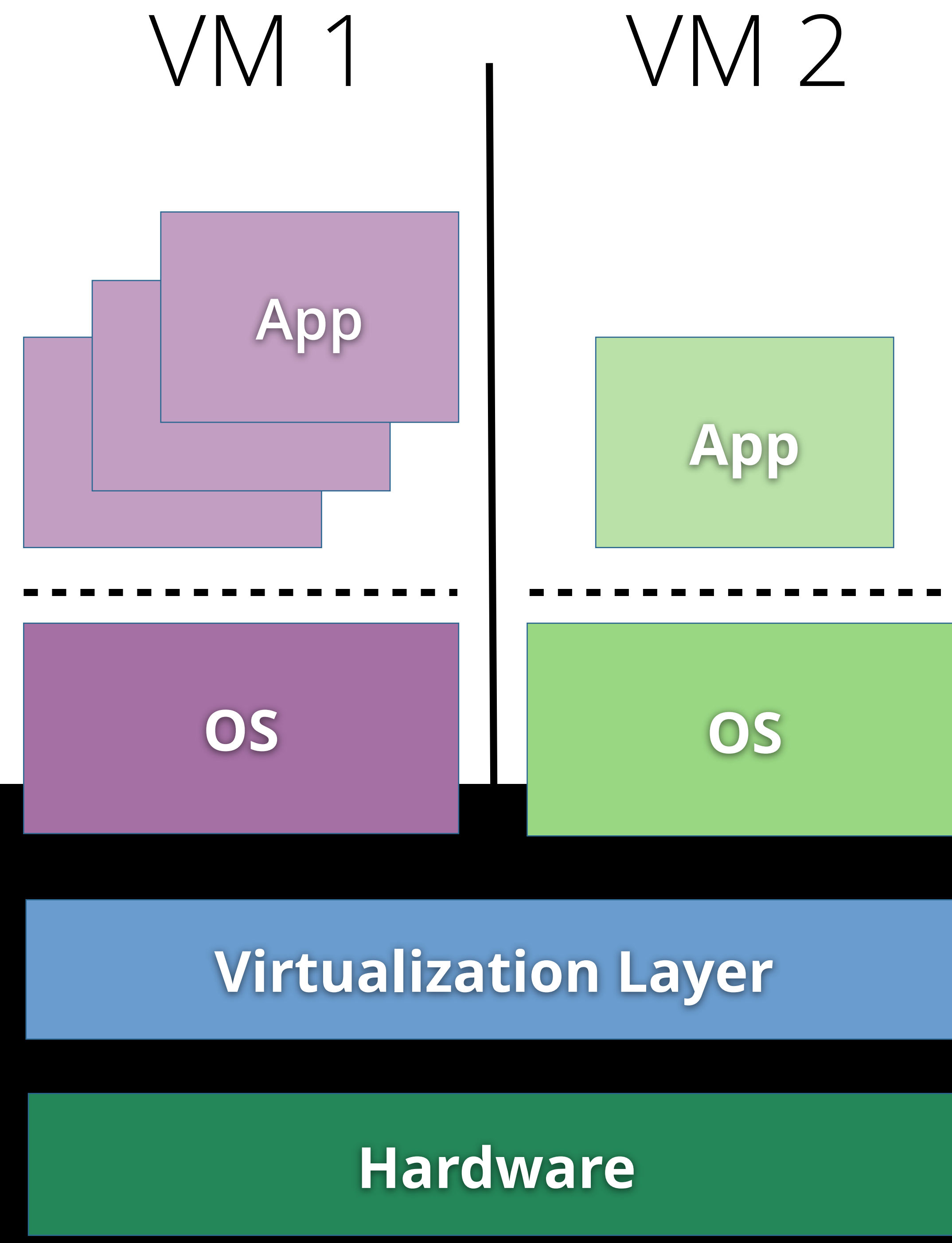
HARDWARE ISOLATION

- Separate computers
- Applications and data physically isolated
- Effective, but …
  - Higher costs
  - [redacted]



VM-BASED ISOLATION

- Multiple VMs, OSes
- Isolation enforced by virtualization layer
- Saves space, energy, maintenance effort
- But still ...



TECHNISCHE UNIVERSITÄT DRESDEN

WHAT IS THE PROBLEM?

- Huge code bases remain
- Applications still the same
- Many targets to attack:
  - Applications, libraries, commodity OSes
  - Virus scanner, firewall, ...

TECHNISCHE UNIVERSITÄT DRESDEN

SECURITY ARCHITECTURES

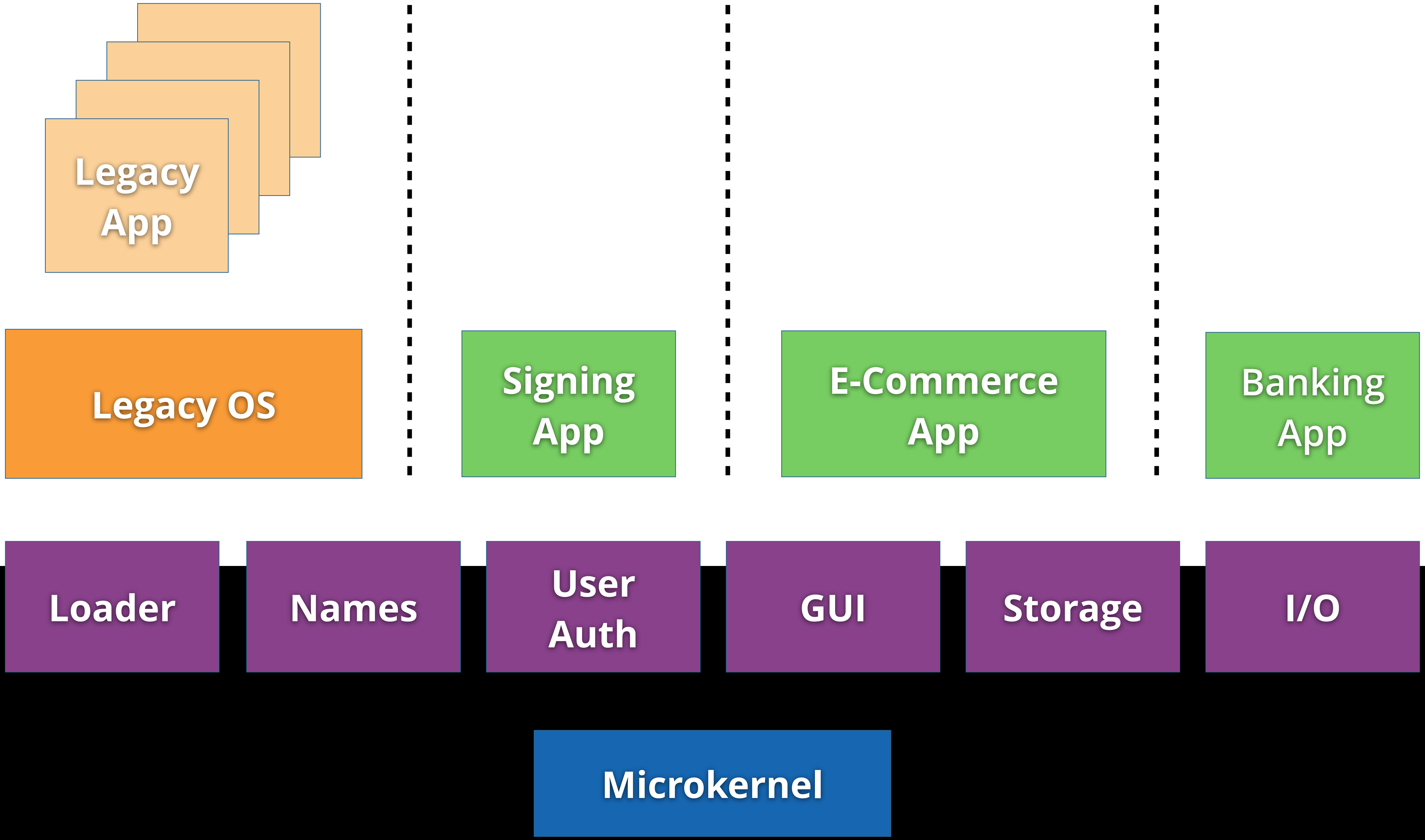
SECURITY GOALS

- Protect the user's data
- Secure applications that process data
- Acknowledge different kinds of trust, e.g.:
  - Application **A** trusted to handle its own data, but not the files of application ***B***
  - [redacted]

APPROACH

- To improve security: Reduce size of TCB = smaller attack surface
- First (incomplete) idea:
  - Remove huge legacy OS from TCB
  - Port application to microkernel-based multi-server OS

NIZZA ARCHITECTURE



NIZZA ARCHITECTURE

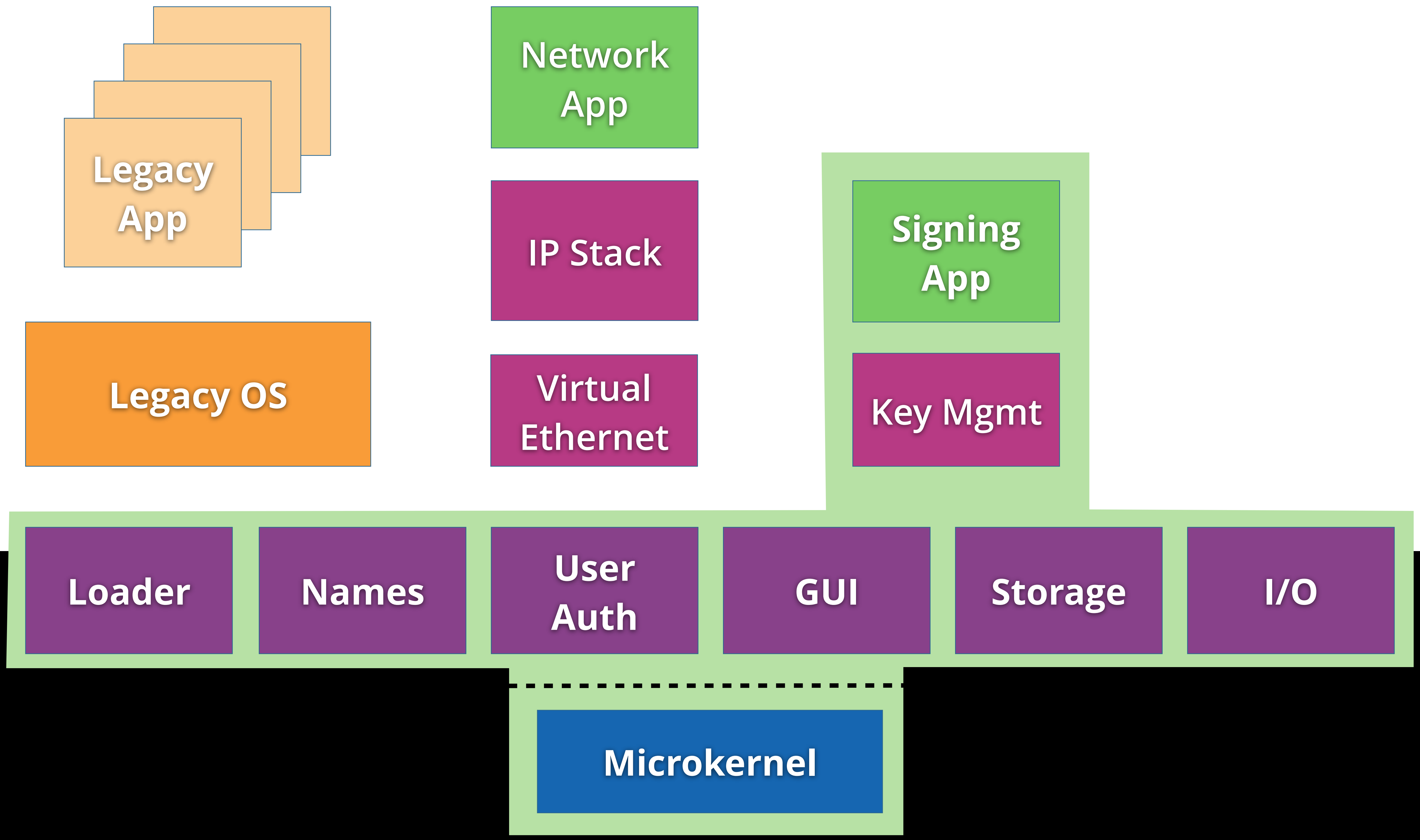
Nizza architecture: fundamental concepts:

- Strong isolation
- Application-specific TCBs
- Legacy reuse
- Trusted computing

APP-SPECIFIC TCB

- Reflects **Principle of Least Privilege**
- TCB of an application includes only components its security relies upon
- TCB does not include unrelated applications, services, libraries

APP-SPECIFIC TCB



APP-SPECIFIC TCB

- Reflects **Principle of Least Privilege**
- TCB of an application includes only components its security relies upon
- TCB does not include unrelated applications, services, libraries
- Mechanisms:

TECHNISCHE UNIVERSITÄT DRESDEN

SPLITTING COMPONENTS

SPLIT APPS

- Problems with porting applications:
  - Dependencies need to be satisfied
  - Can be complex, require lots of code
  - Stripped down applications may lack functionality / usability
- Better idea: split application

EXAMPLE 1: EMAIL

Digitally signed e-mails, what's critical?

- Handling of signature keys
- Requesting passphrase to unlock signature key
- Presenting e-mail message:
  - [redacted]

STANDARD EMAIL APP

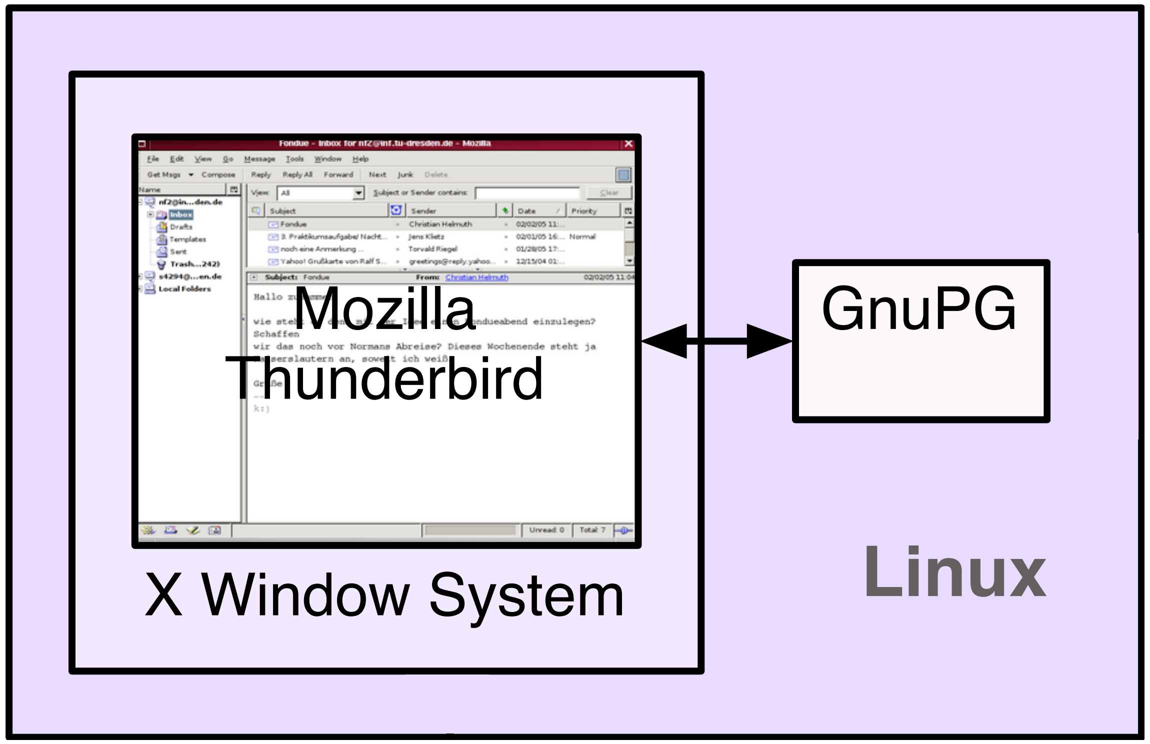


Image source: [5]

SPLIT EMAIL APP

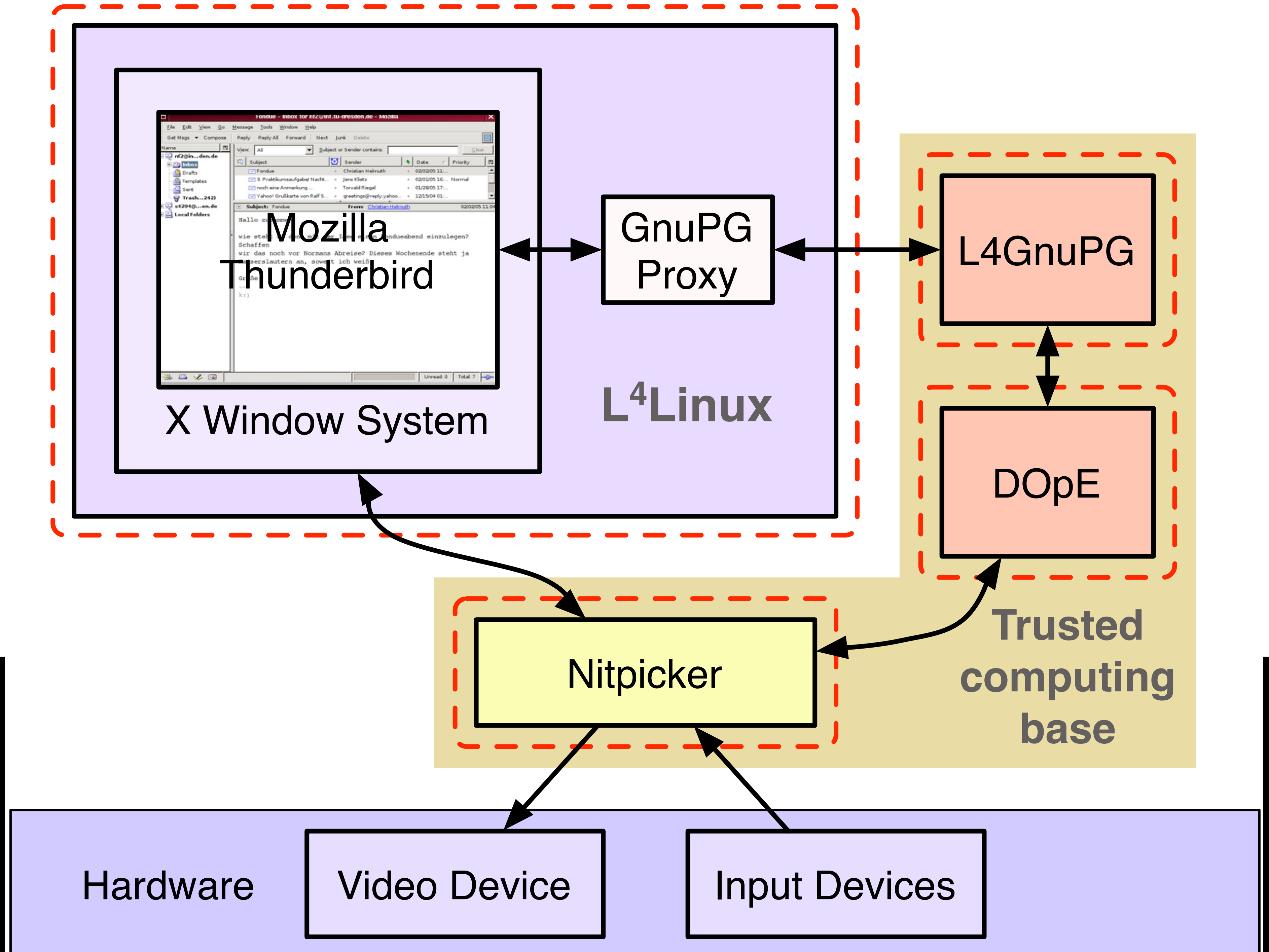


Image source: [5]

TCB REDUCTION

- *1,500,000*+ SLOC no longer in TCB:
  - Linux kernel, drivers, X-Server
  - C and GUI libraries, Thunderbird, …
- TCB size reduced to ~*150,000* SLOC:
  - GNU Privacy Guard, e-mail viewer

SPLITTING THE OS

- Splitting works for applications
- What about the complex and useful infrastructure of commodity OSes?
  - Drivers (see previous lectures)
  - Protocol stacks (e.g., TCP/IP)

SIMPLE REUSE

- Run legacy OS in VM
- Reuse service: net, files, …
- Legacy infrastructure isolated from applications
- But:
  - [redacted]

COMPLEXITY + BUGS

- Network and file system stacks are virtually essential subsystems
- Generally well tested
- Ready for production use
- … but not bug free [1,2]:

[redacted]

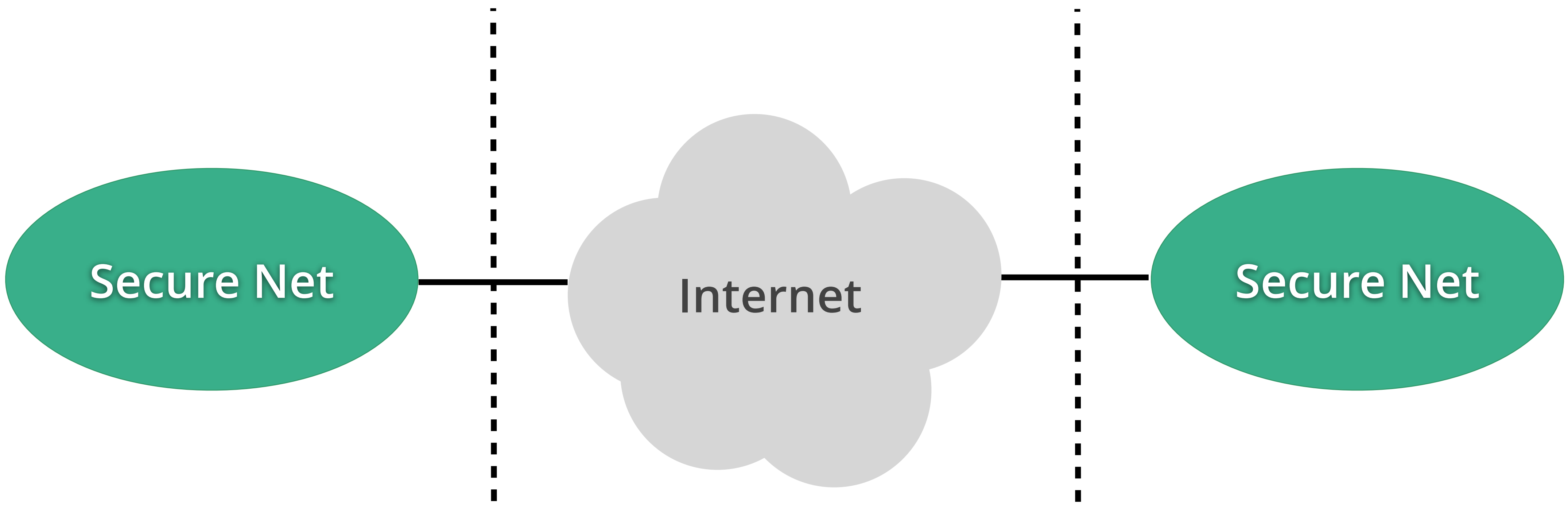
REUSE + TRUST

- Complex protocol stacks should not be part of TCB (for confidentiality + integrity)
- Reuse untrusted infrastructure through **Trusted Wrapper:**
  - Add security around existing APIs
    - Cryptography

TECHNISCHE UNIVERSITÄT DRESDEN

EXAMPLE 2: VPN

VPN: Confidentiality, Integrity, ~~Availability~~



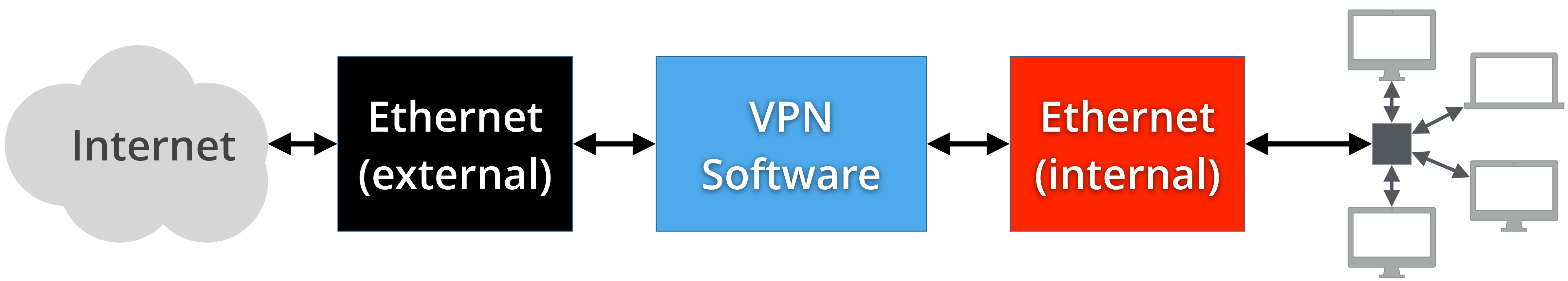
CASE STUDY: SINA BOX

- SINA box used by German „BSI“:
- VPN gateway
- Implements IPSec & PKI
- Intrusion detection & response

Image source:
http://www.secunet.com/de/das-unternehmen/presse/bilddatenbank/

SINA BOX

- Differently trusted network interfaces:
  - **Red:** plaintext, no protection
  - **Black:** encryption + authentication codes



OS COMPLEXITY

- Linux is complex!
- SLOC for Linux 2.6.18:
  - Architecture specific: 817,880
  - x86 specific: 55,463
  - Drivers: 2,365,256

[redacted]

Released date: 20 Sep 2006

OS COMPLEXITY

- Linux is even more complex in 2024!
- SLOC for Linux 6.7.1:
  - Architecture specific: 1,729,519
  - x86 specific: 316,544
  - Drivers: 17,771,667

[redacted]

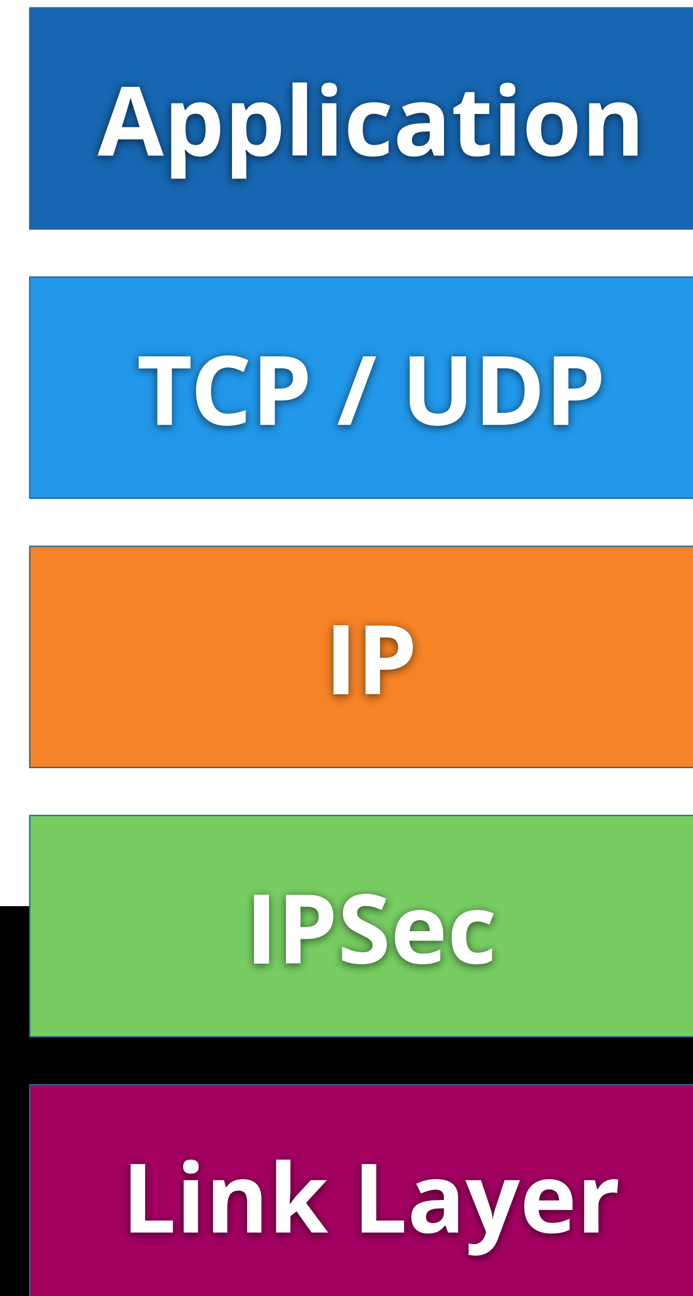
MIKRO-SINA

- Research project „Mikro-SINA“
- Goals:
  - Reduce TCB of VPN gateway software
  - Enable high-level evaluation for high assurance scenarios
  - [redacted]

IPSEC BASICS

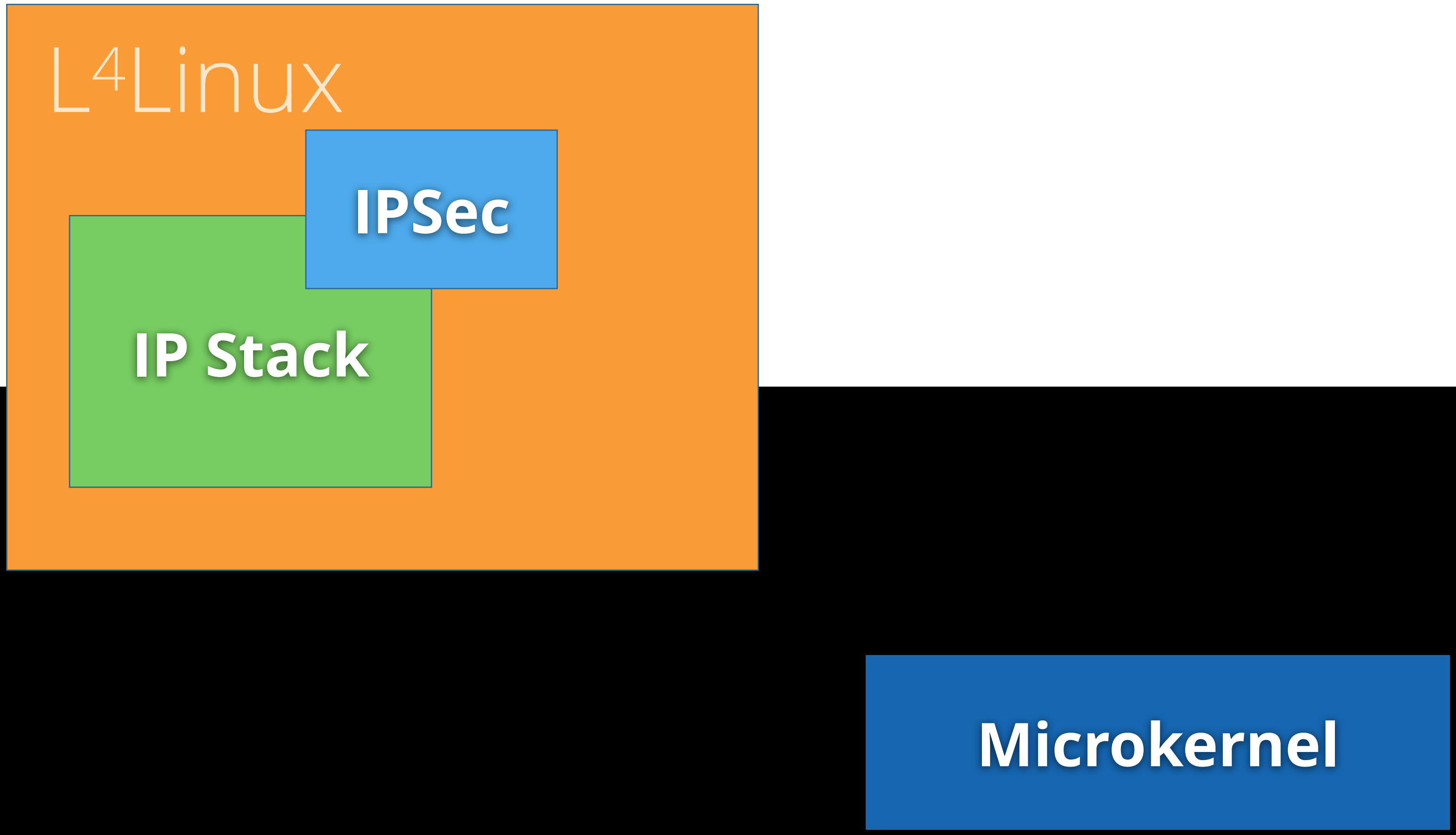
- Protocol suite for securing IP-based communication
- Authentication header (**AH**)
  - Integrity
  - Authentication
- Encapsulating Security Payload

[redacted]



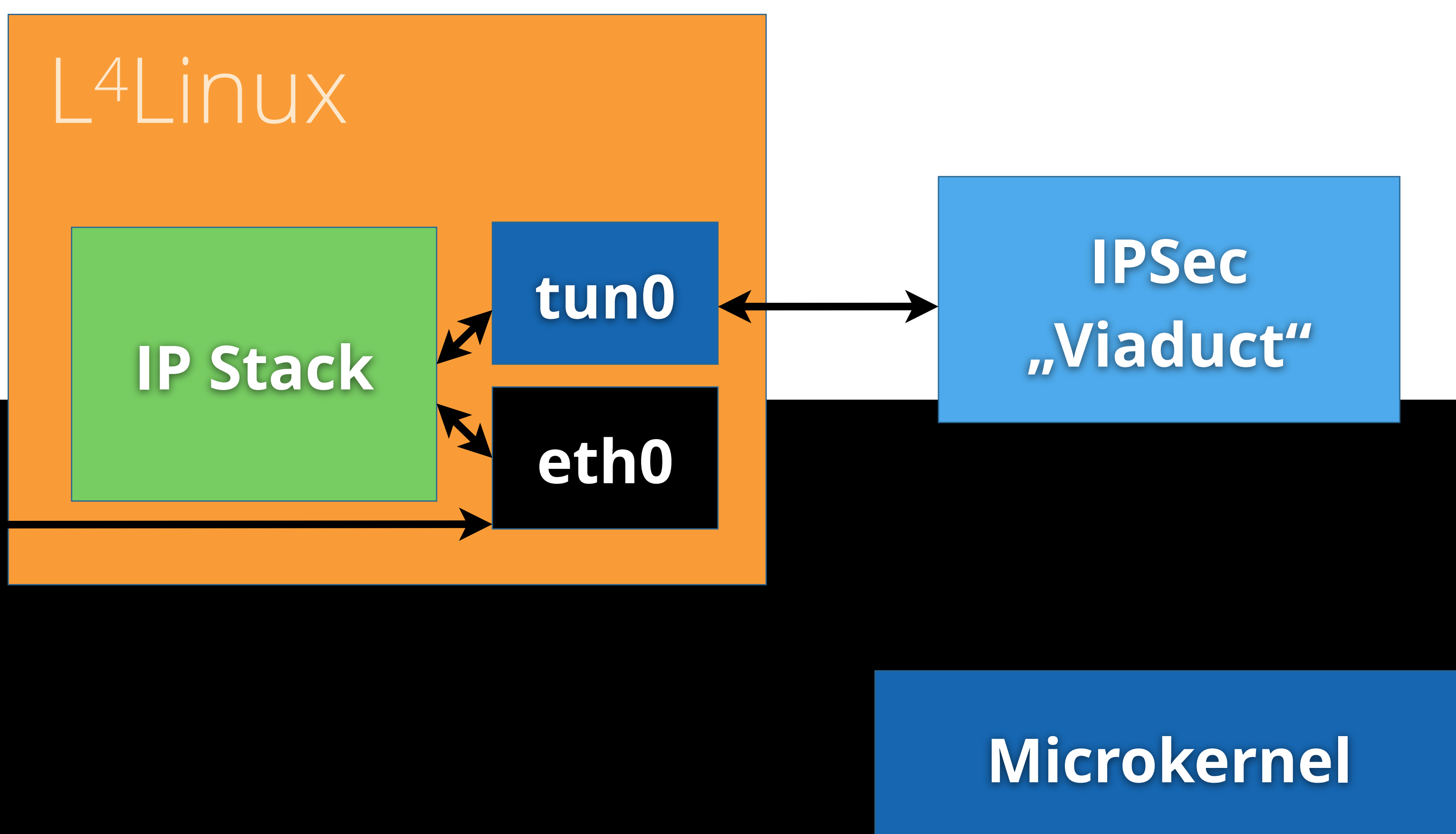
IPSEC IN L⁴LINUX

- IPSec is security critical component
- … but is integrated into Linux kernel



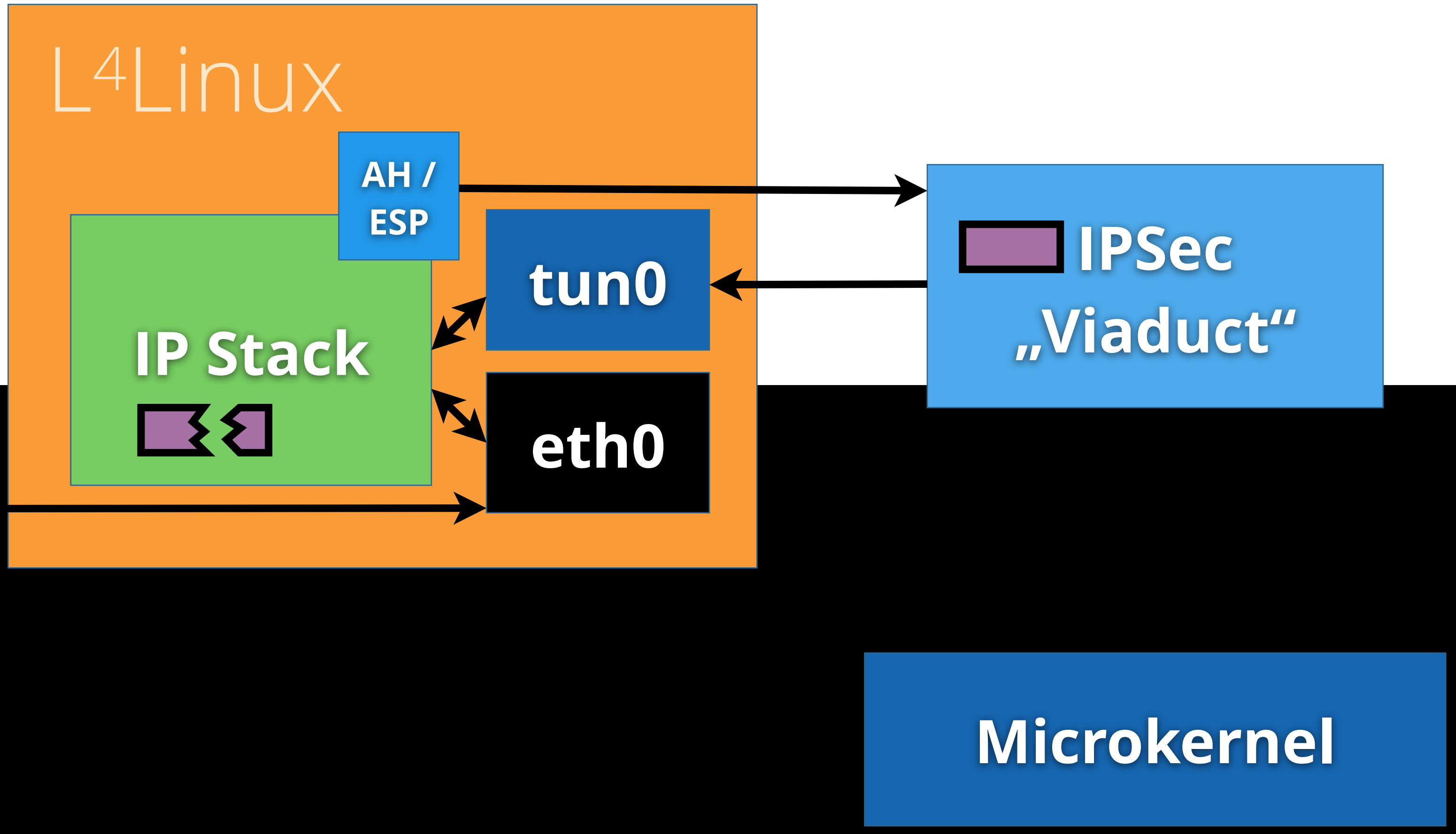
IPSEC „VIADUCT"

- **Idea:** Isolate IPSec in „Viaduct"
- IPSec packets sent/received through TUN/TAP device



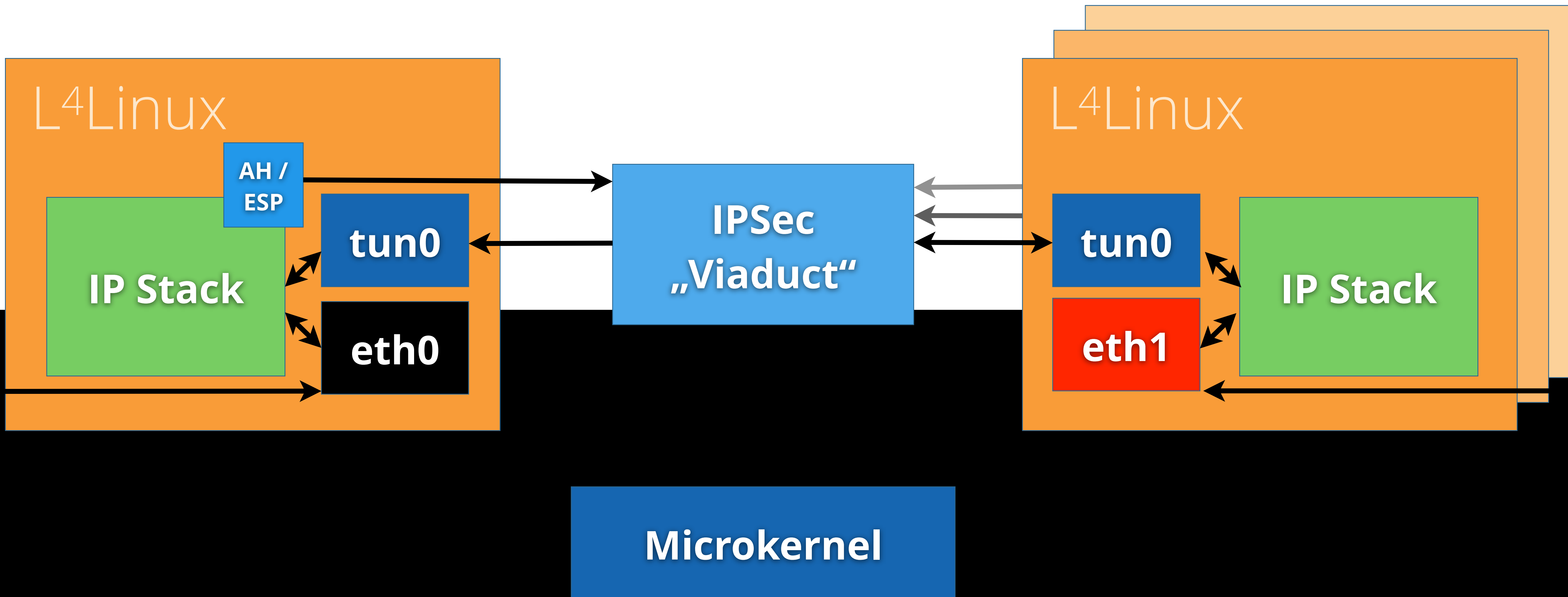
FRAGMENTATION

- Problem: Routers can fragment IPSec packets on the way
- Let L4Linux reassemble them



CONFIDENTIALITY

- Untrusted L4Linux instances must not see both plaintext and encrypted data
- Dedicated L4Linux for black/red networks



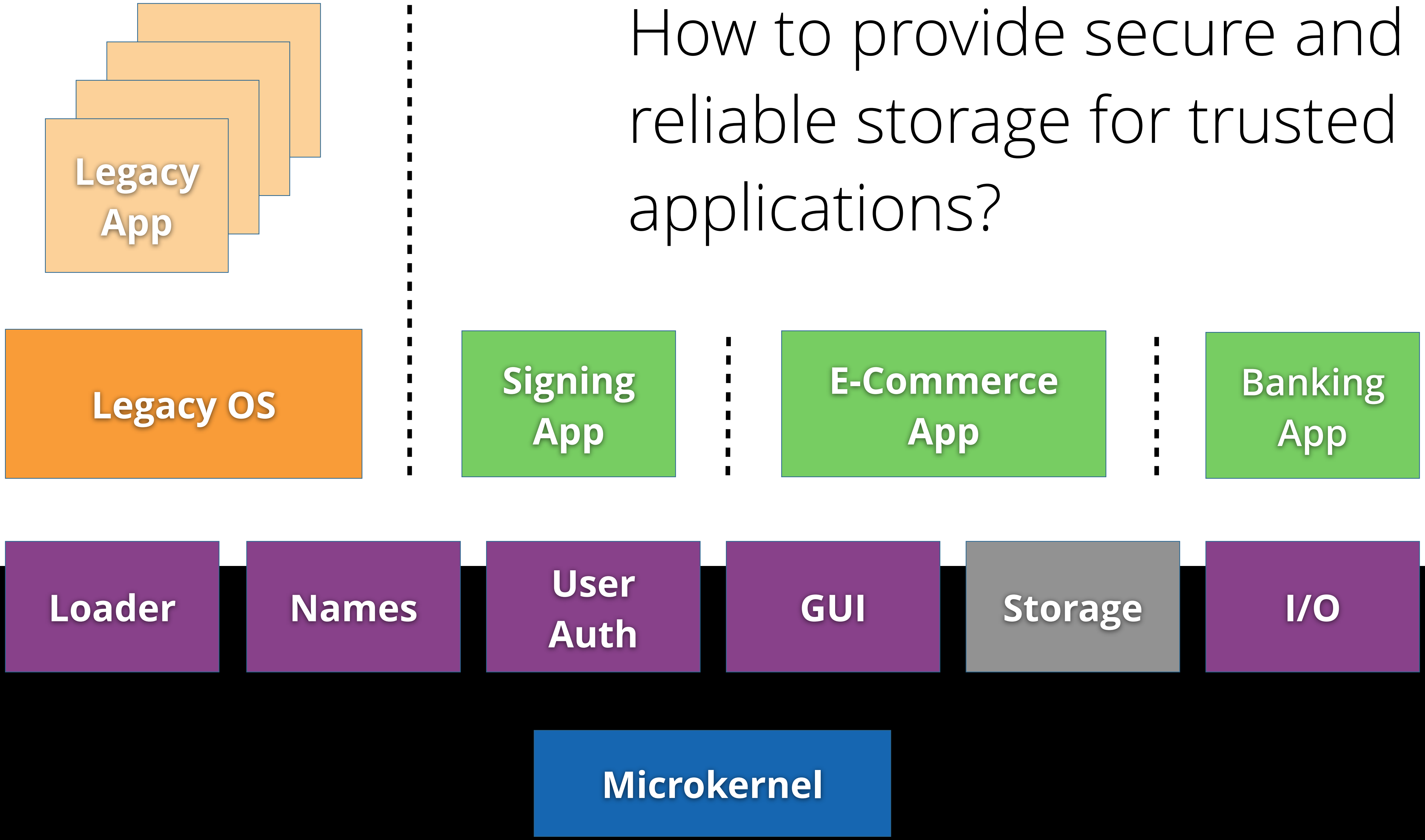
MIKRO-SINA

- Result: Trusted Wrapper for VPN
- Small TCB (see [6] for details):
  - 5,000 SLOC for „Viaduct“
  - Fine grain isolation
  - Principle of least privilege

[redacted]

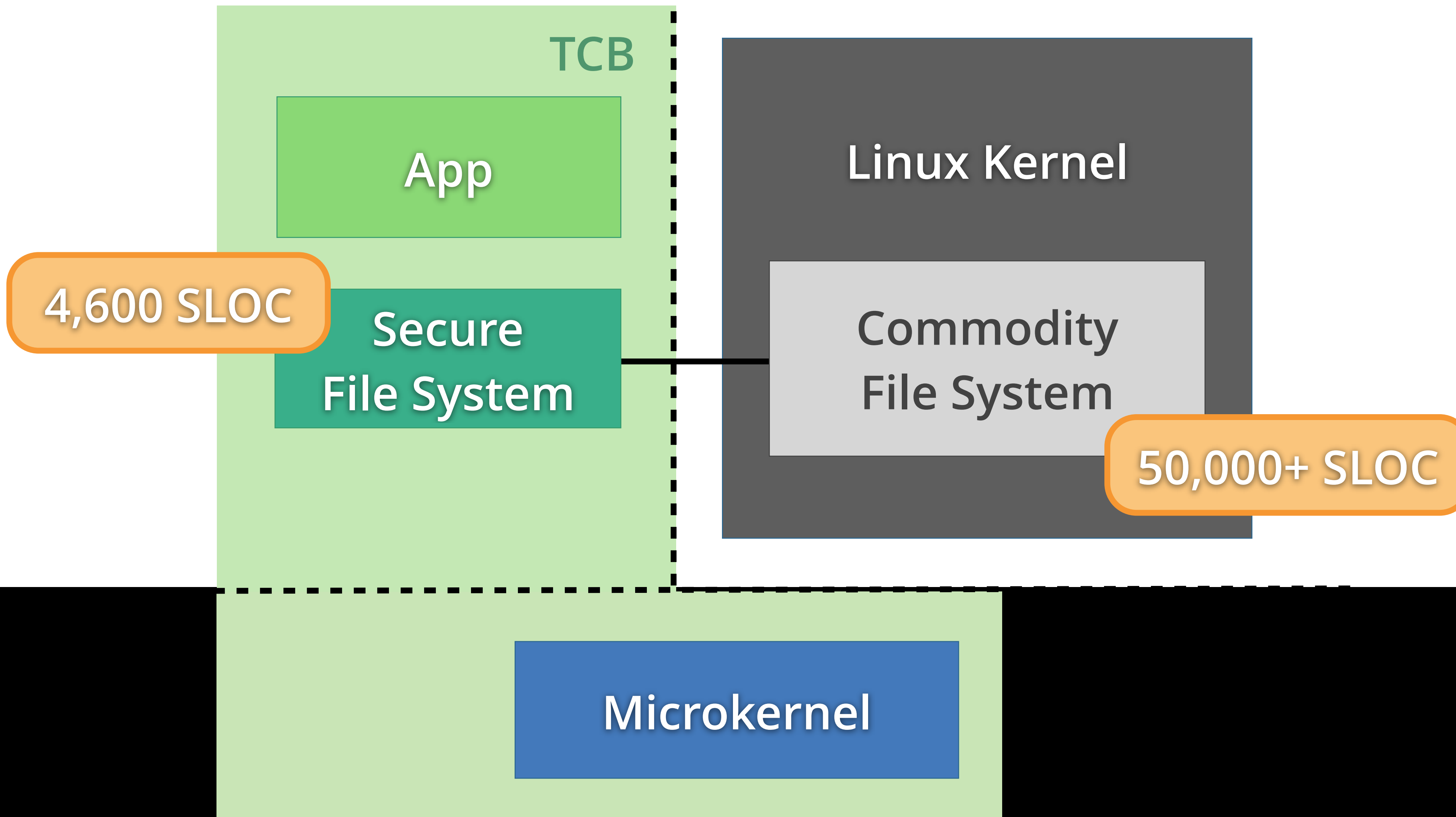
EXAMPLE 3: STORAGE

How to provide secure and reliable storage for trusted applications?

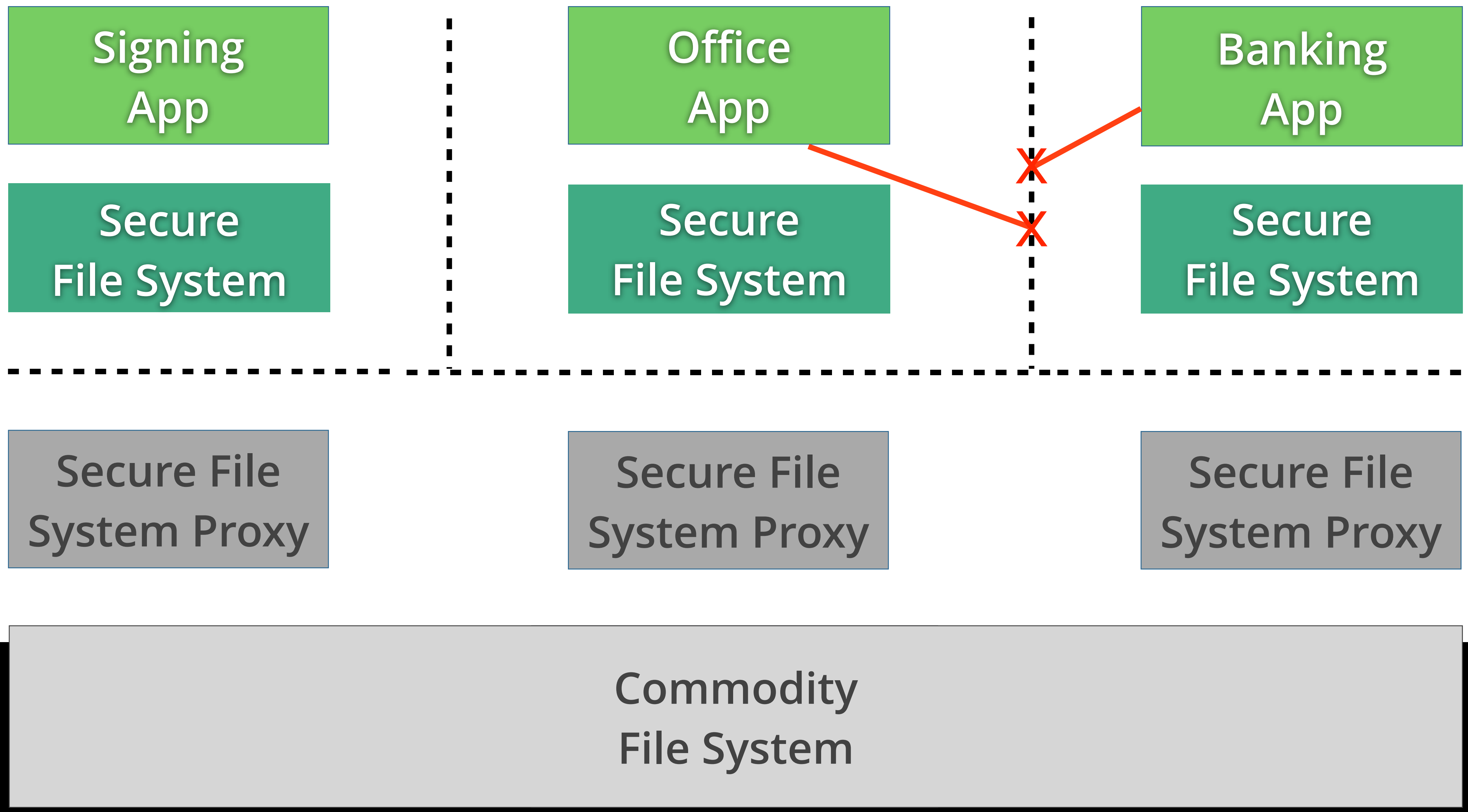


VIRTUAL PRIVATE...

VPFS: Confidentiality, Integrity, ~~Availability~~



VPFS STACK

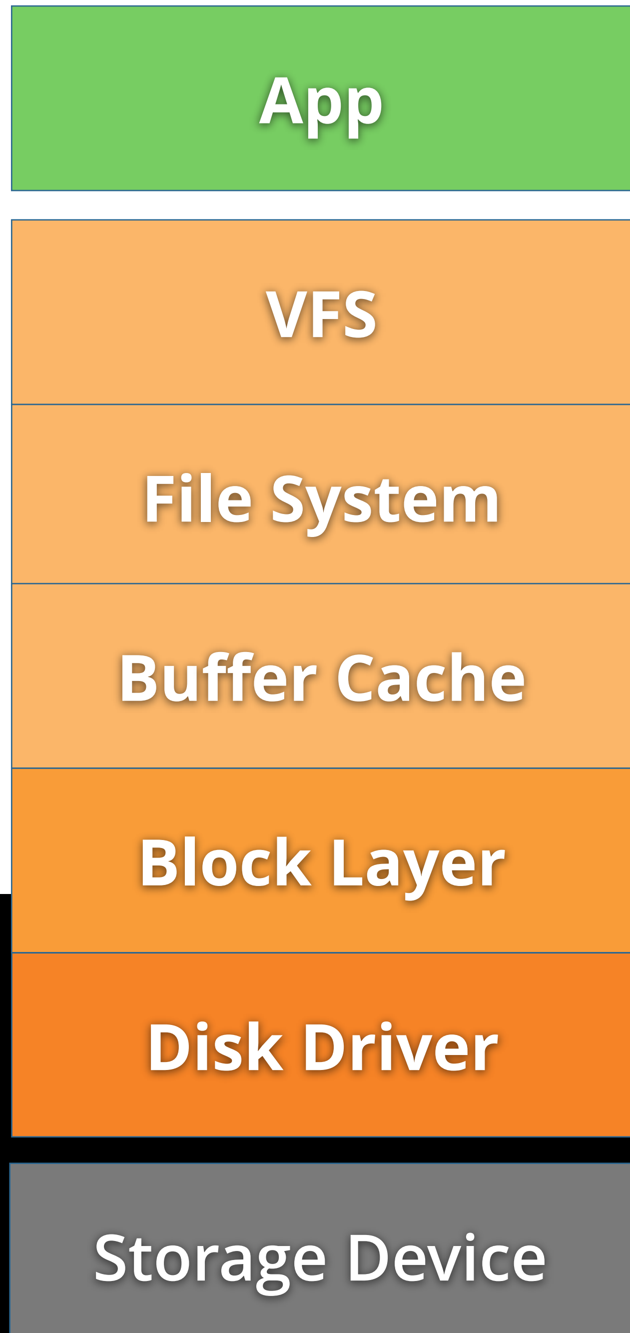


Signing App
Office App
Banking App
Secure File System
Secure File System
Secure File System
Secure File System Proxy
Secure File System Proxy
Secure File System Proxy
Commodity File System

SECURITY GOALS

- **Confidentiality:** only authorized applications can access file system, all untrusted software cannot get any useful information
- **Integrity:** all data and meta data is correct, complete, and up to date;
- **Recoverability:** damaged data in untrusted file system can be recovered

POPULAR SOLUTIONS



File-level protection

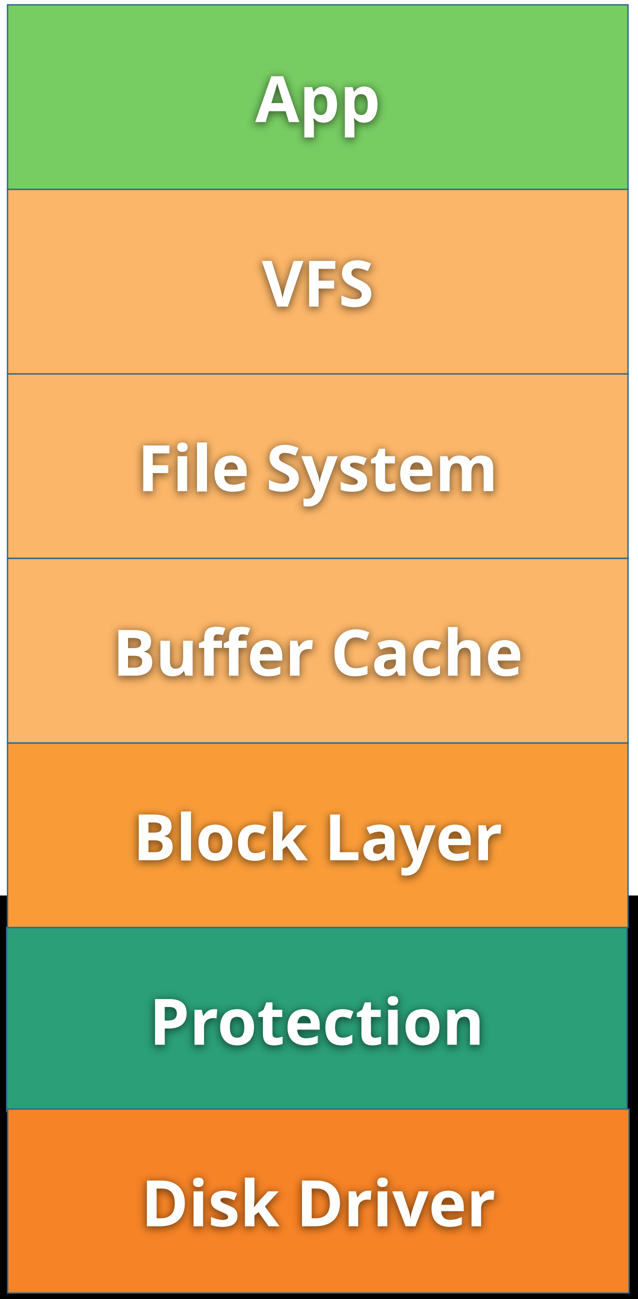
CFS	Cryptographic File System for UNIX
EFS	Microsoft Encrypting File System
ecryptfs	Linux kernel support + tools
EncFS	Based on FUSE

Volume-level protection

TrueCrypt, **Filevault 2**
dm_crypt
Bitlocker
Encrypted volumes in smartphones, etc.

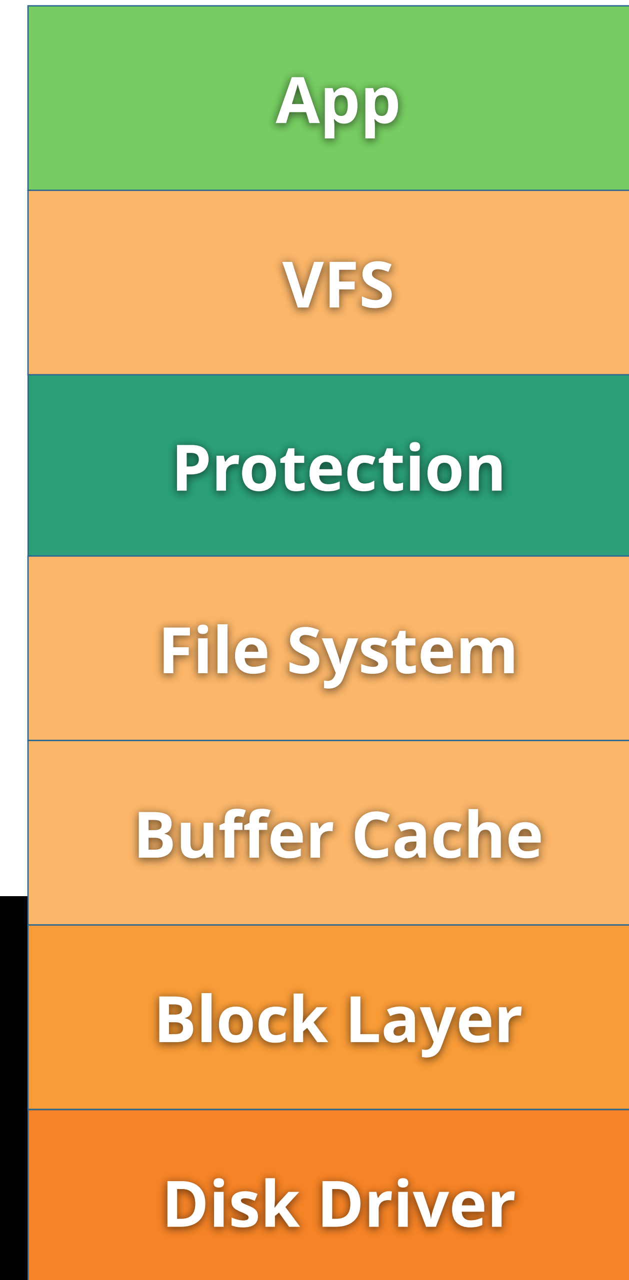
DESIGN OPTIONS

- First end of design space: Protect at block layer
  - Transparent encryption of all data and metadata
  - Block-level integrity ???
  - [redacted]

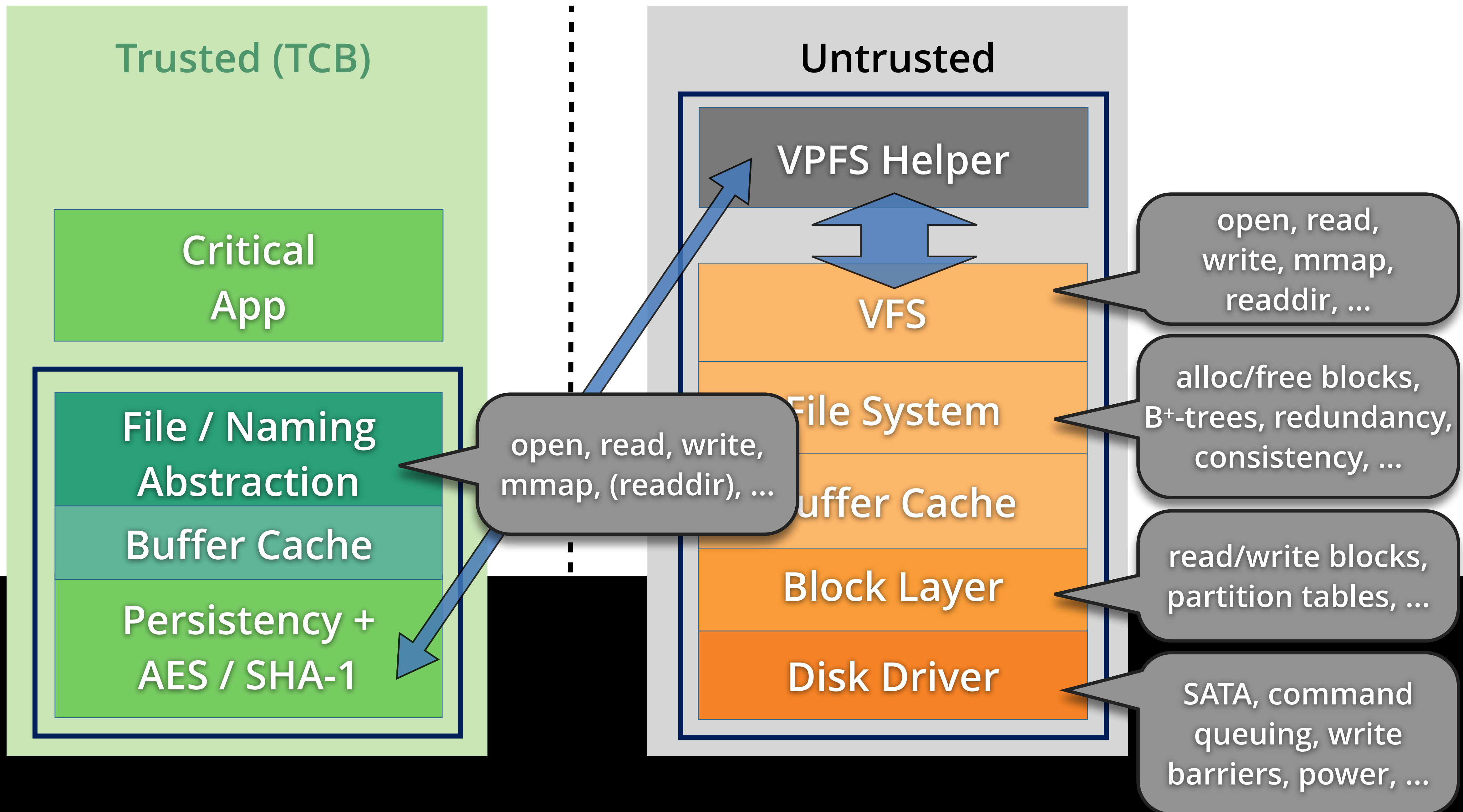


DESIGN OPTIONS

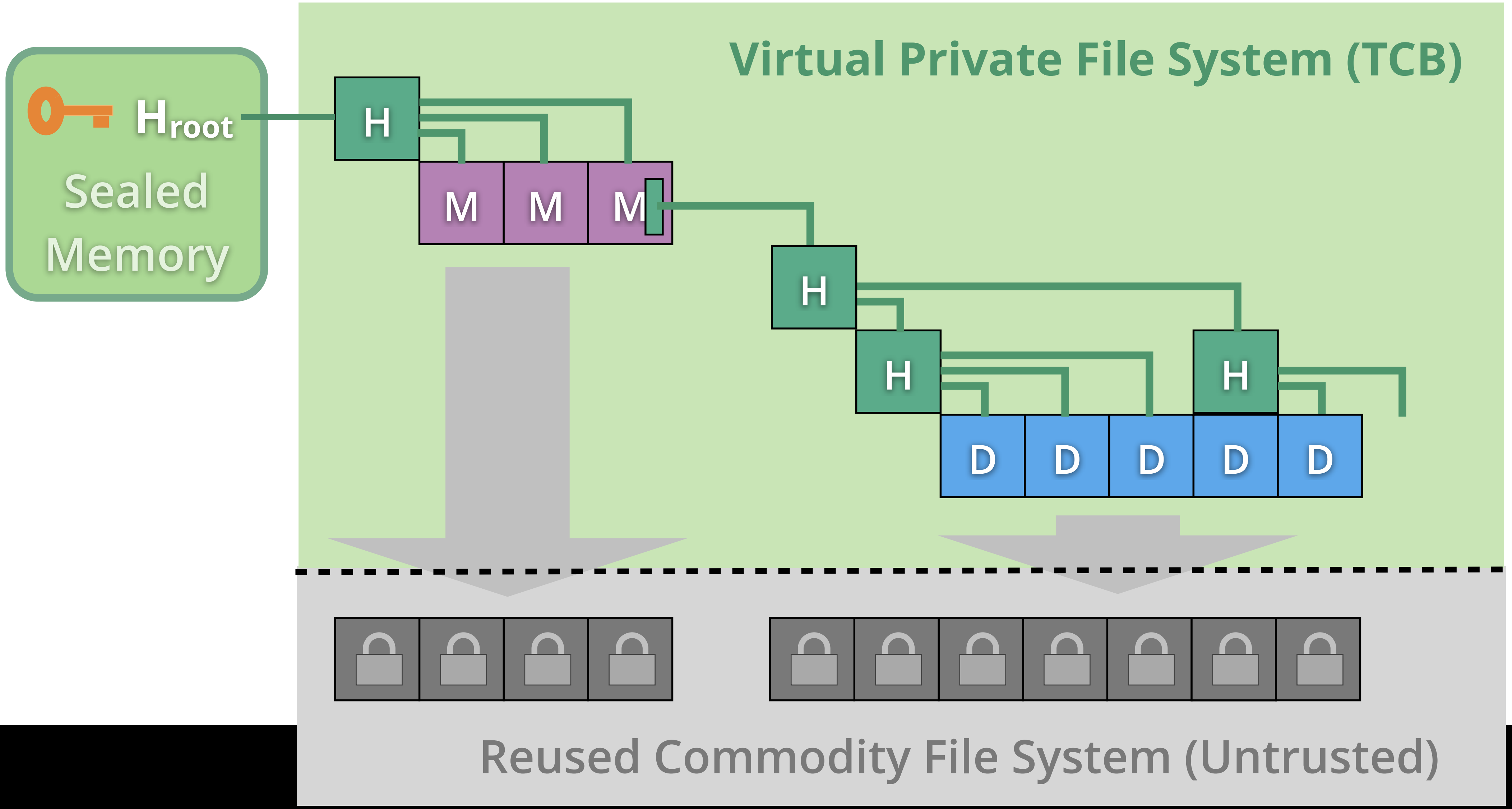
- Second end of design space: Protect individual files
  - Stacked file system
  - Encrypt all data and some metadata (directories, …)
  - More flexibility for integrity



TRUSTED WRAPPER



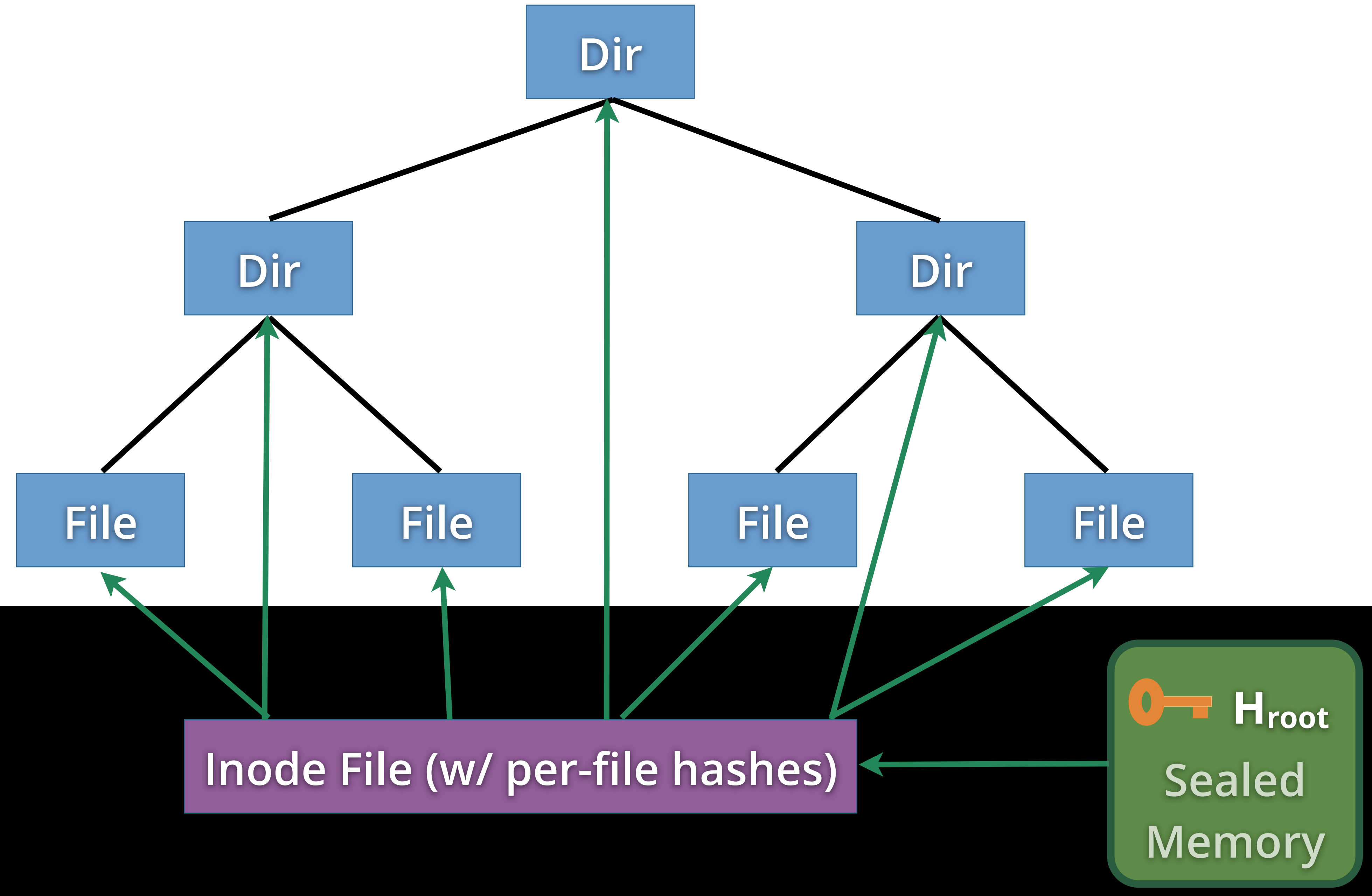
VPFS APPROACH



VPFS APPROACH

- Trusted part of VPFS enforces security:
  - Encryption / decryption on the fly
  - Plaintext only in trusted buffer cache
  - Files in untrusted commodity file system store encrypted blocks
  - [redacted]

MULTIPLE FILES



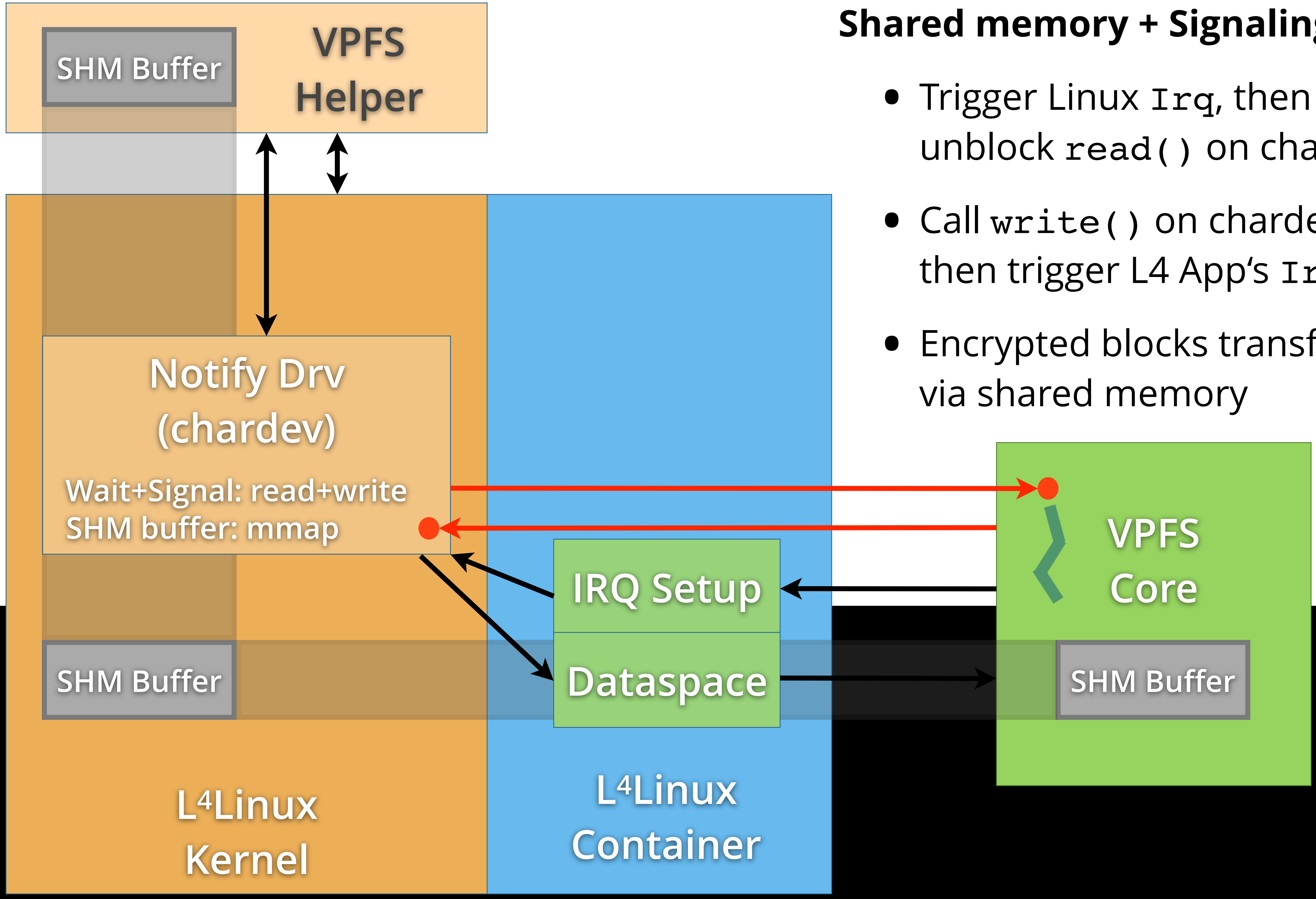
TECHNISCHE UNIVERSITÄT DRESDEN

EXTENSIVE REUSE

- VPFS reuses Linux file system stack:
  - Drivers, block device layer
  - Optimizations (buffer cache, read ahead, write batching, ...)

[redacted]

VPFS PROXY DRIVER



Shared memory + Signaling:

- Trigger Linux `Irq`, then unblock `read()` on chardev
- Call `write()` on chardev, then trigger L4 App's `Irq`
- Encrypted blocks transferred via shared memory

VPFS SUMMARY

- Trusted wrappers for file systems work!
- VPFS is general purpose file system
- Significant reduction in code size:
  - Untrusted Linux file system stack comprises **50,000+** SLOC
  - [redacted]

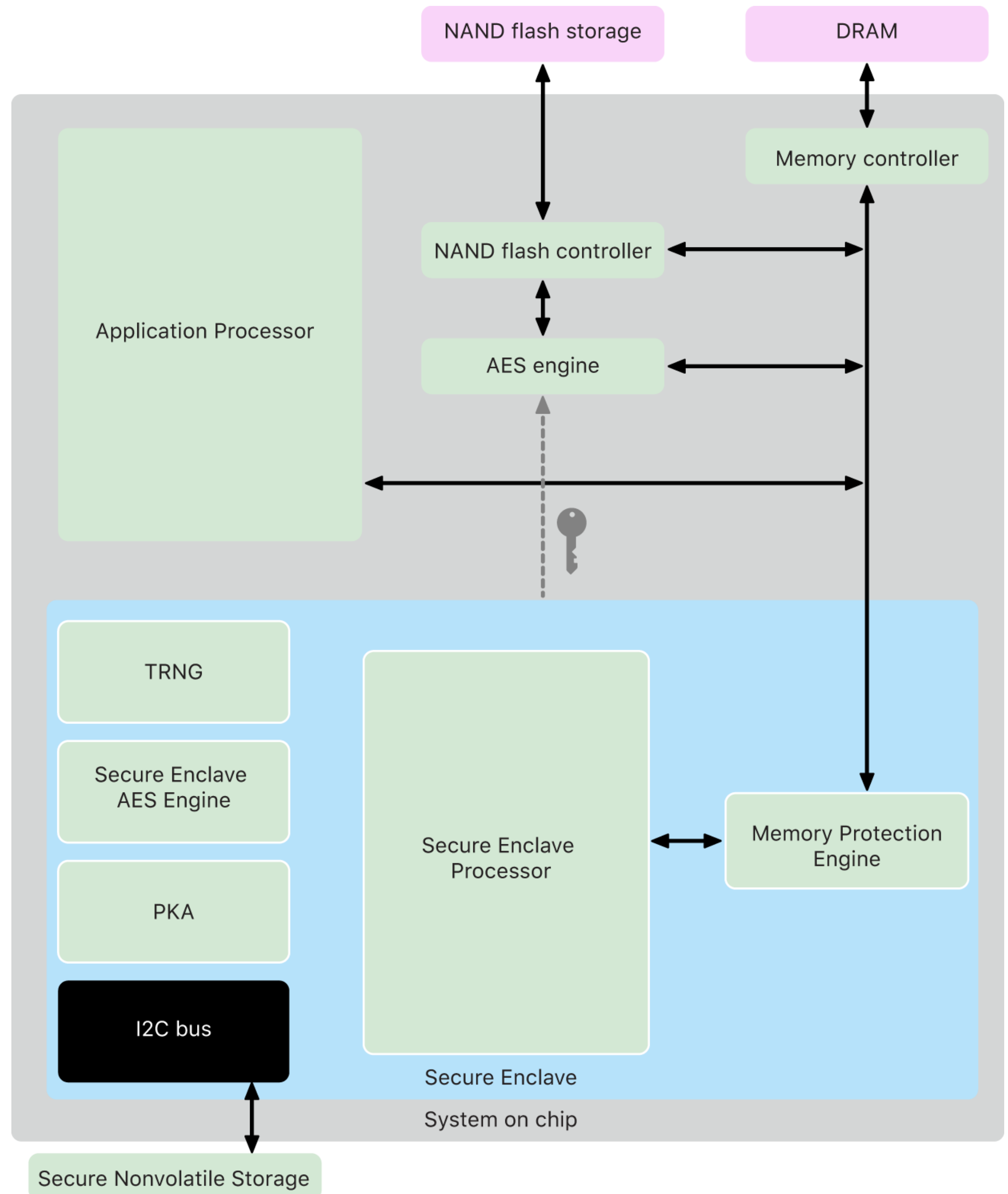
SUMMARY

- Secure reuse of untrusted legacy infrastructure
- Split apps + OS services for smaller TCB
- Nizza secure system architecture:
  - Strong isolation

[redacted]

BRIEFLY: HARDWARE ISOLATION

APPLE SECURE ENCLAVE

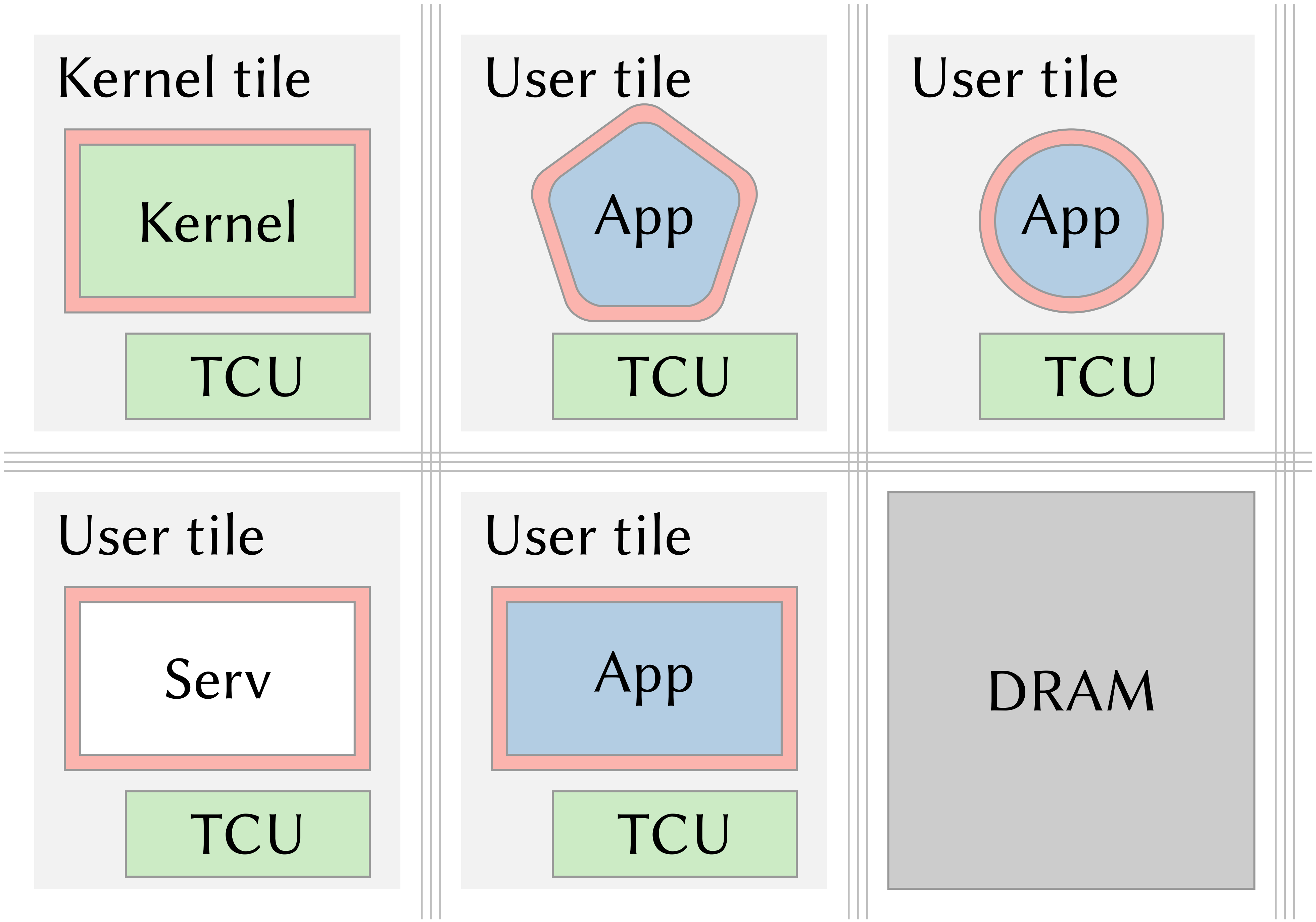


Apple devices have "Secure Enclave Processor (SEP)" running a dedicated service OS fully isolated from from the application processor hardware.

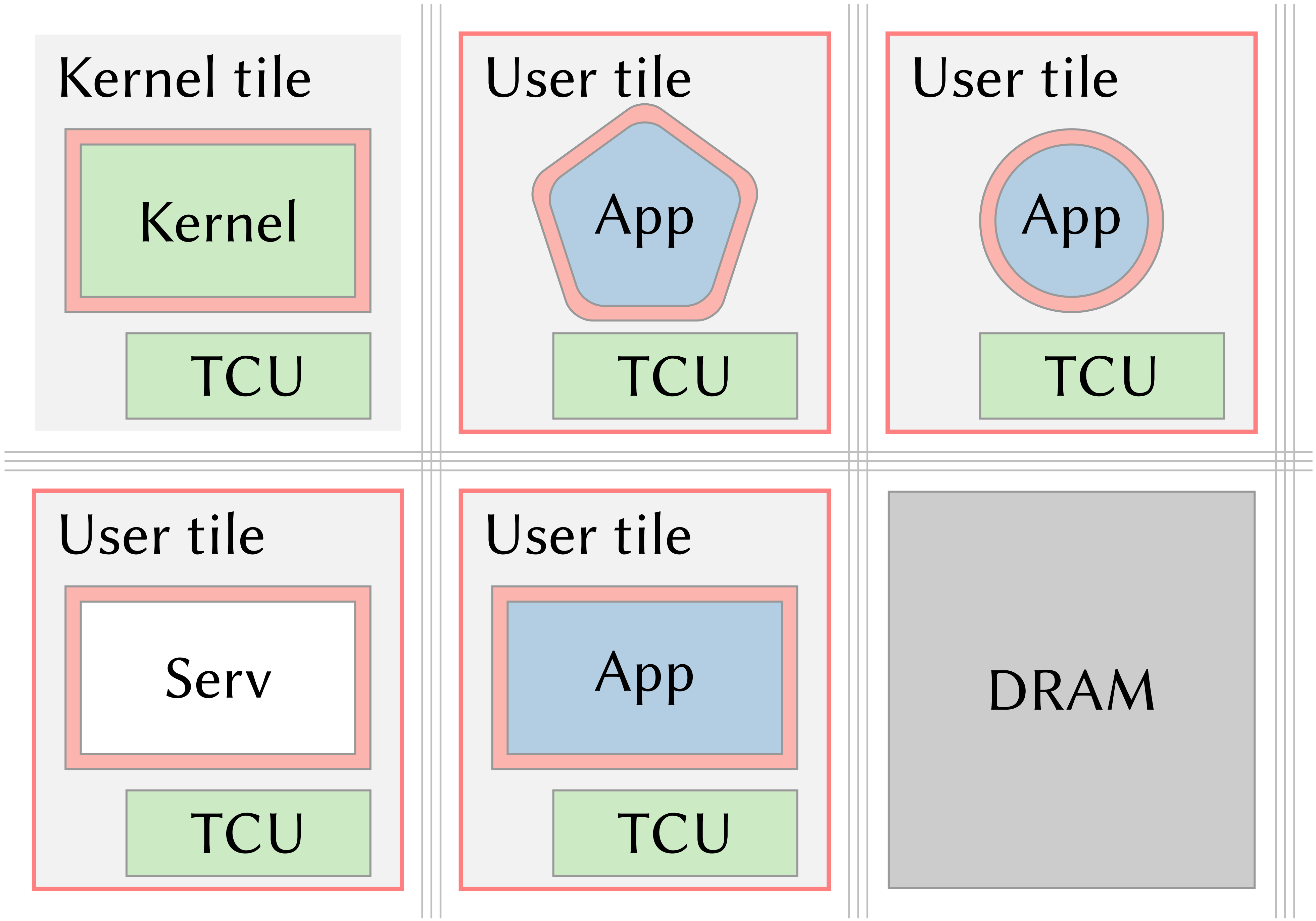
[redacted]

Source: Apple Support Documentation
https://support.apple.com/guide/security/secure-enclave-sec59b0b31ff/web

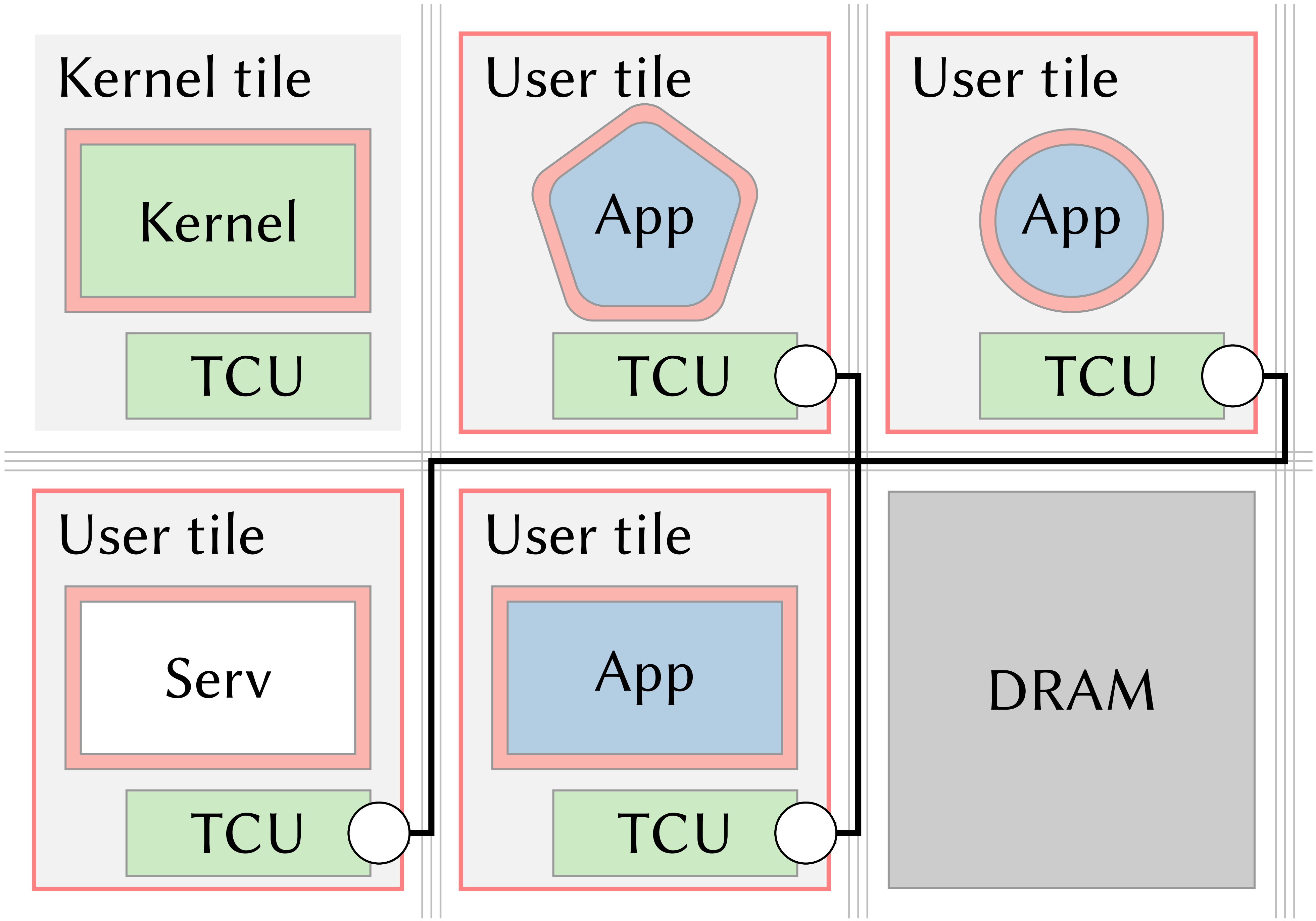
SNEAK PEEK: M³



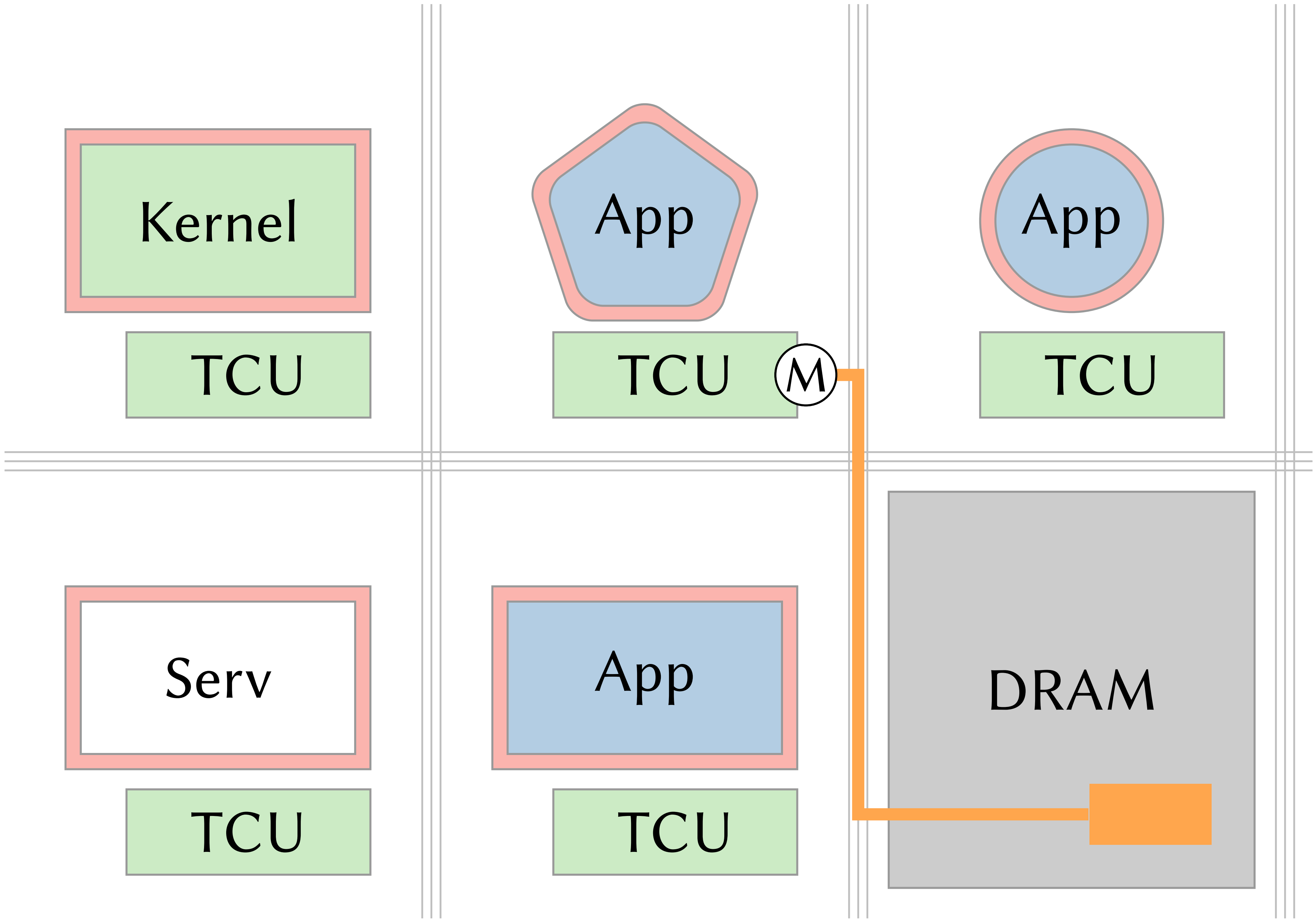
SNEAK PEEK: M^3



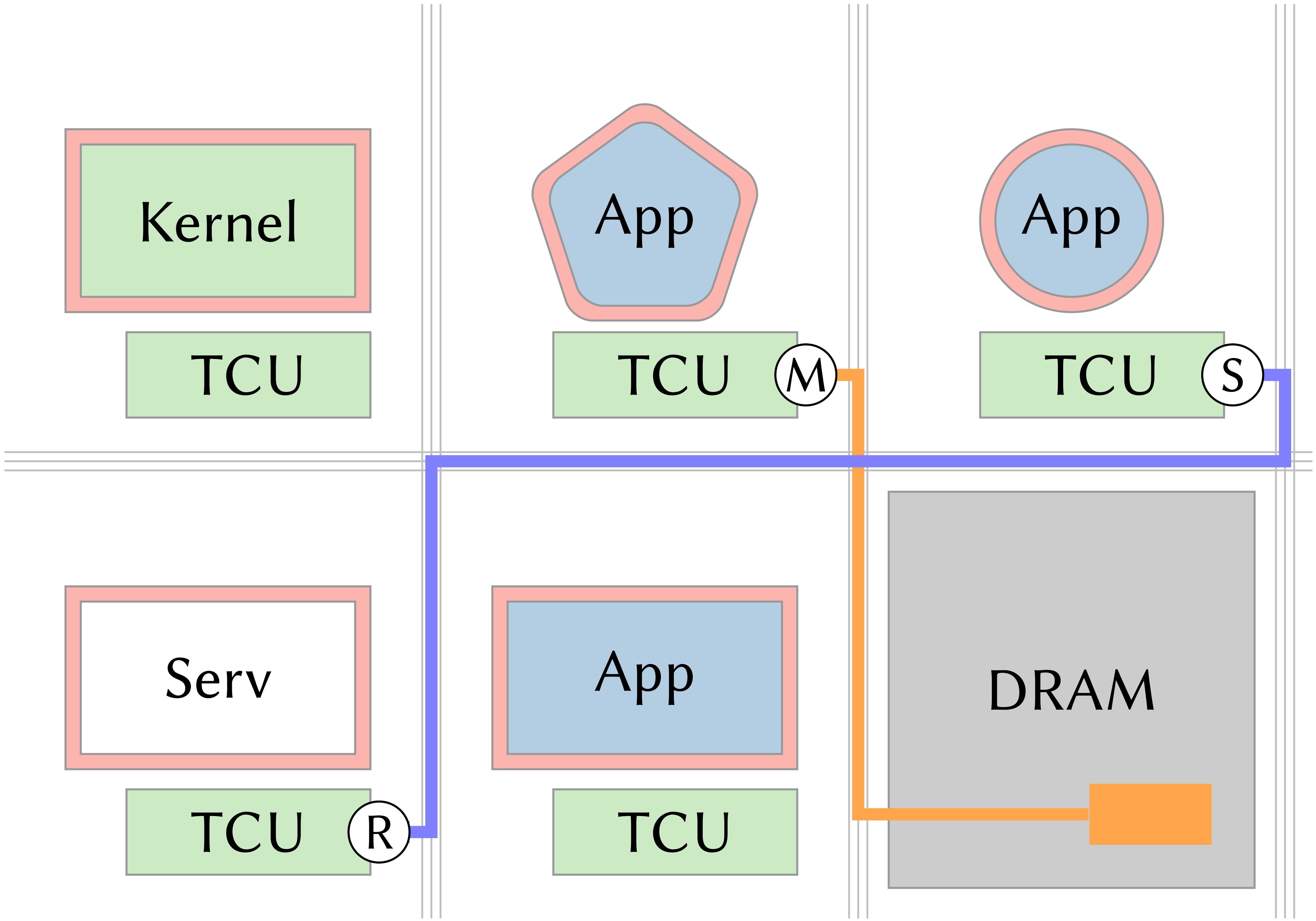
SNEAK PEEK: M³



SNEAK PEEK: M^3



SNEAK PEEK: M³



REFERENCES

- [1] http://www.heise.de/newsticker/Month-of-Kernel-Bugs-Ein-Zwischenstand--/meldung/81454
- [2] http://projects.info-pull.com/mokb/
- [3] Carsten Weinhold and Hermann Härtig, „VPFS: Building a Virtual Private File System with a Small Trusted Computing Base“, Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems, April 2008, Glasgow, Scotland UK
- [4] Carsten Weinhold and Hermann Härtig, „jVPFS: Adding Robustness to a Secure Stacked File System with Untrusted Local Storage Components“, Proceedings of the 2011 USENIX Annual Technical Conference, Portland, OR, USA, June 2011
- [5] Norman Feske and Christian Helmuth, „A Nitpicker's guide to a minimal-complexity secure GUI“, ACSAC '05: Proceedings of the 21st Annual Computer Security Applications Conference, 2005, Washington, DC, USA
- [6] Christian Helmuth, Alexander Warg, Norman Feske, „Mikro-SINA - Hands-on Experiences with the Nizza Security Architecture“, D.A.CH Security 2005, 2005, Darmstadt, Germany