seL4 Overview

Principles, Abstractions, Use

Gernot Heiser
Trustworthy Systems @ UNSW Sydney

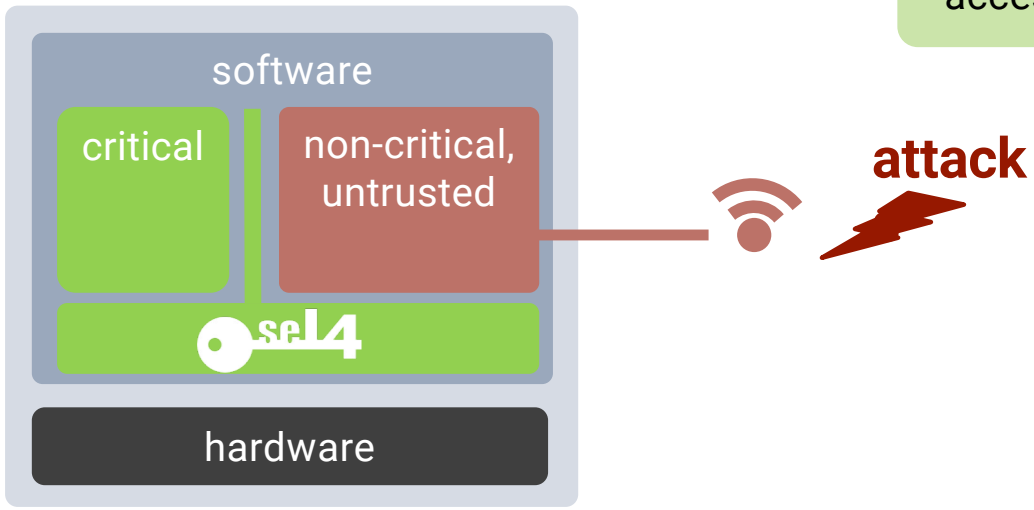
gernot@sel4.systems

What Is seL4?

What is seL4?

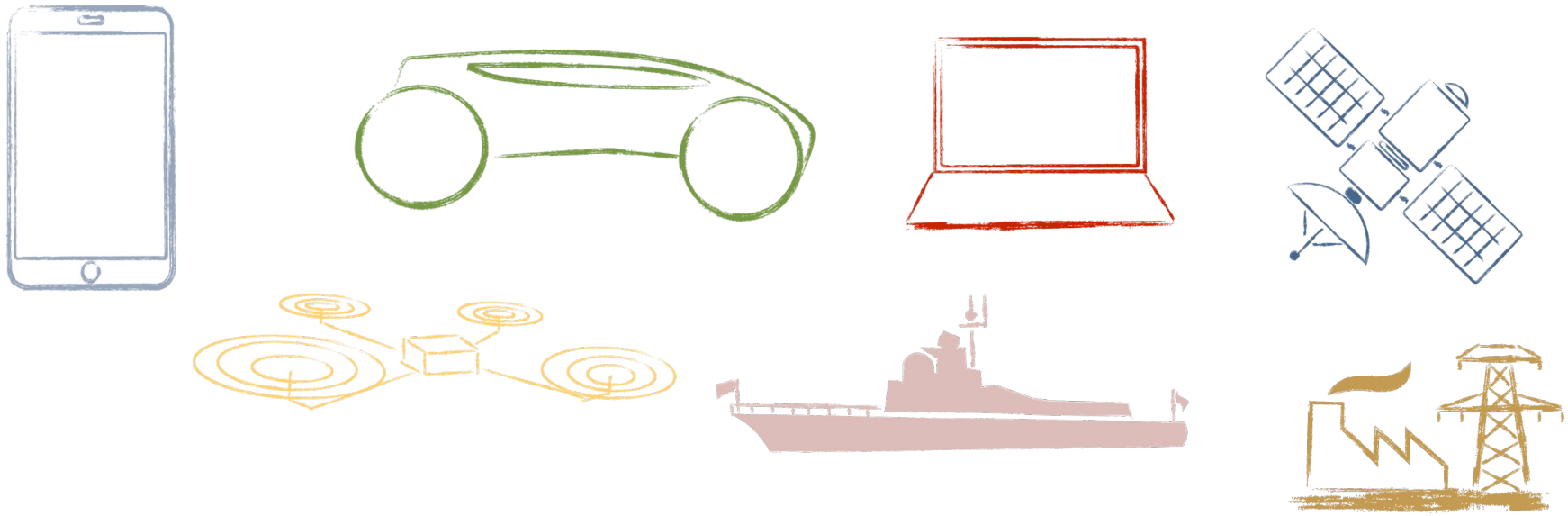
seL4 is an open source, high-assurance, high-performance operating system microkernel

- Available on GitHub under GPLv2 license
- World's most comprehensive mathematical proofs of correctness and security
- World's fastest microkernel
- Piece of software that runs at the heart of any system and controls all accesses to resources



What is seL4?

seL4 is the most trustworthy foundation for safety- and security-critical systems

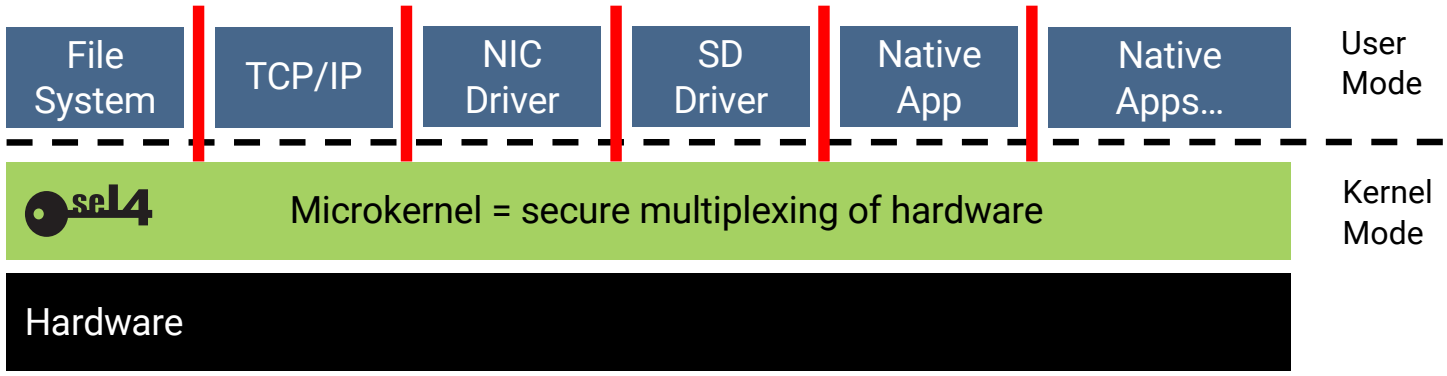


Already in use across many domains:
automotive, aviation, space, defence, critical infrastructure, cyber-physical systems, IoT, industry 4.0, certified security...

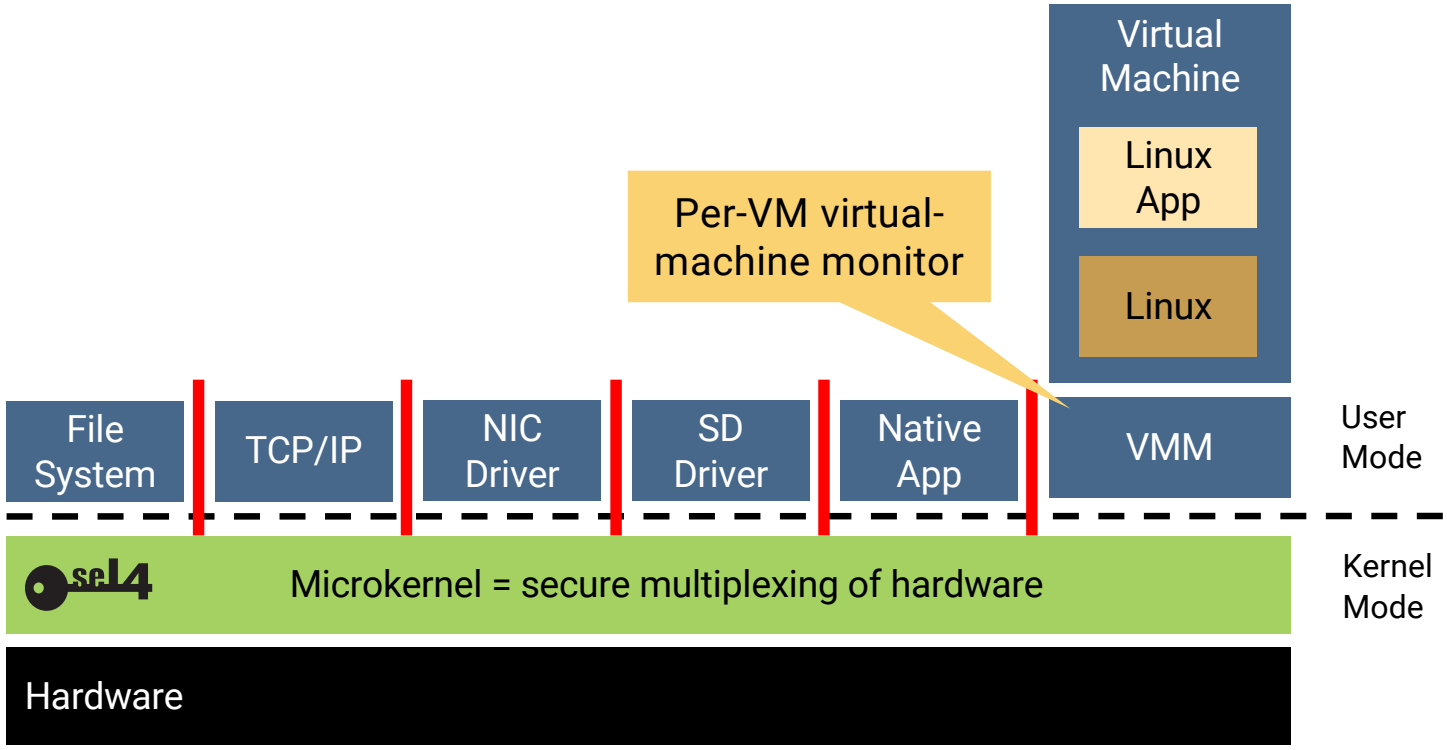
It’s a Microkernel – Not an Operating System

All operating-system services are user-level processes:

- file systems
- device drivers
- power management
- …

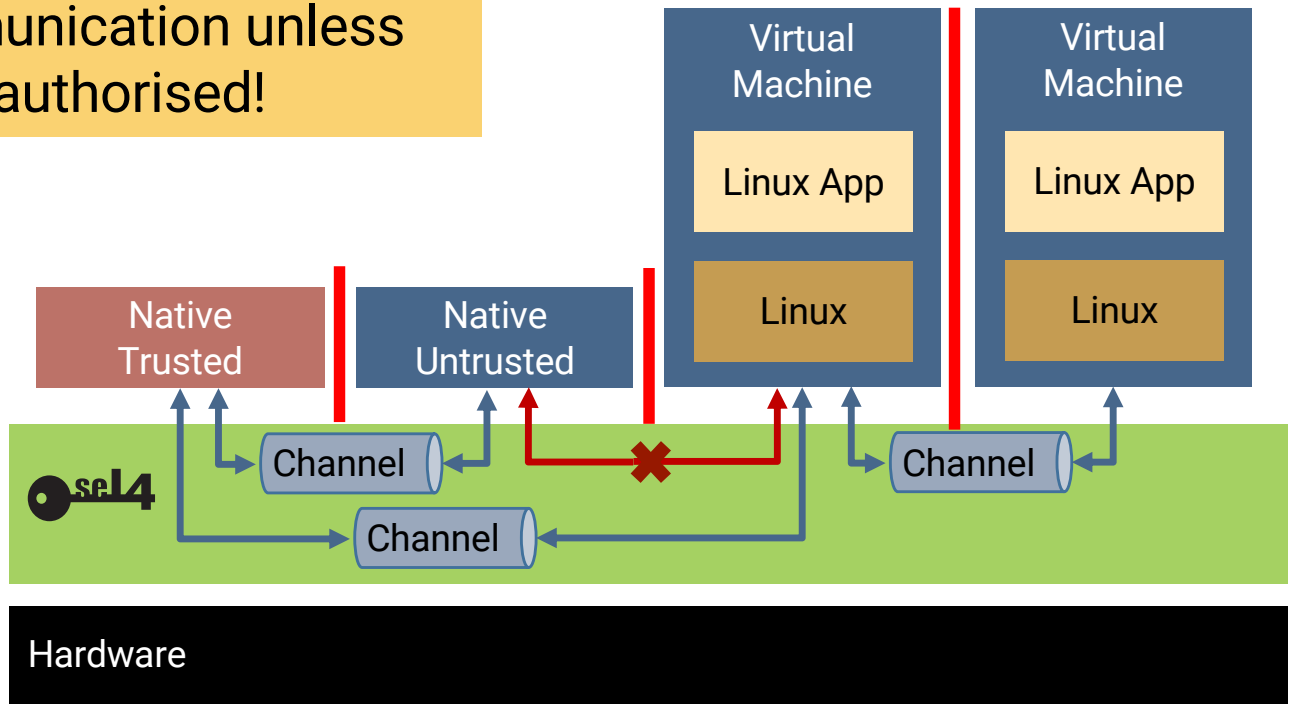


It’s Also a Hypervisor



Capabilities: Fine-Grained Access Control

- Enforce *least privilege*
- No communication unless explicitly authorised!



The Benchmark for Performance

Latency (in cycles, **small is good**) of a round-trip, cross-address-space IPC on x64

World's fastest microkernel!

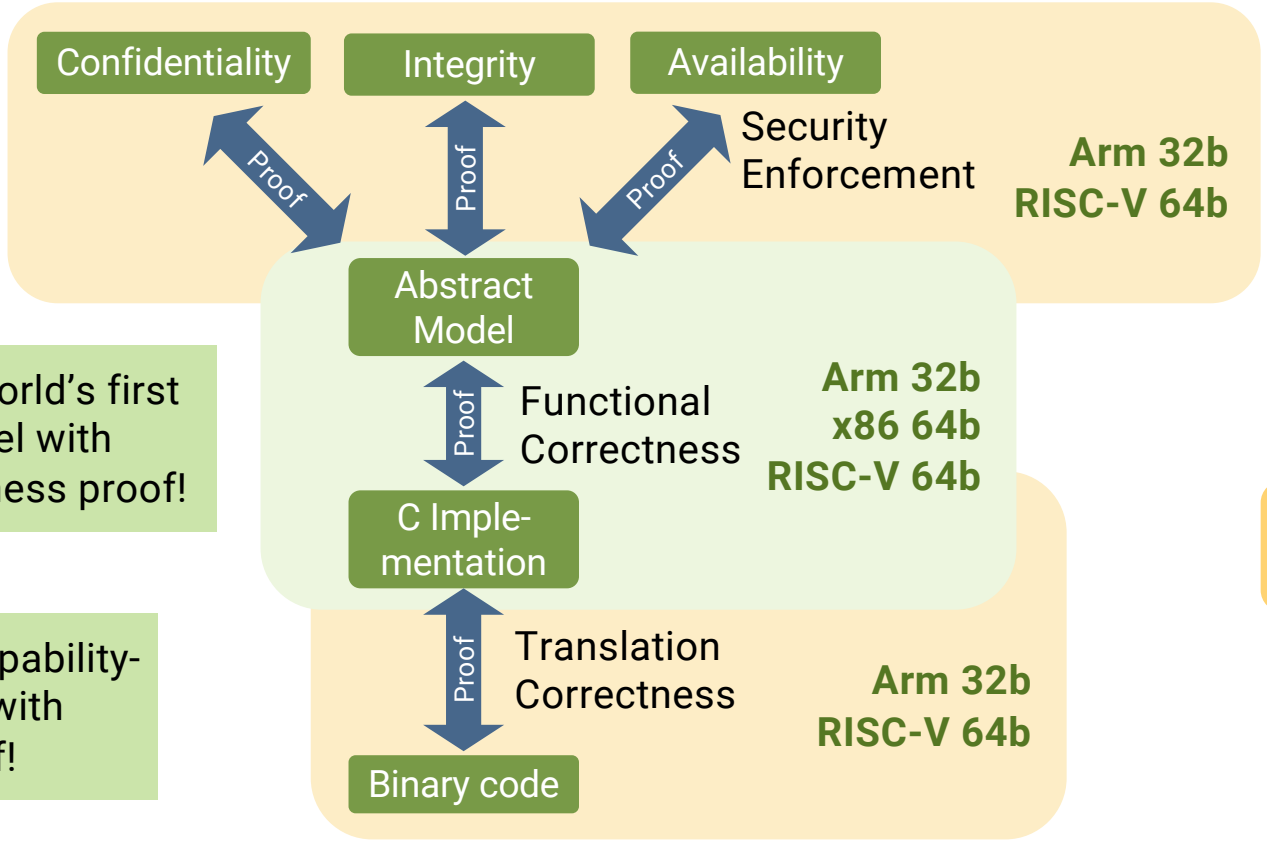
Source	**seL4**	Fiasco.OC	Zircon
Mi et al, 2019	**986**	2717	8157
seL4.systems, Oct'22	**764**	– –	– –

Within 10–20% of hardware limit!

Sources:

- Zeyu **Mi**, Dingji Li, Zihan Yang, Xinran Wang, Haibo Chen: "SkyBridge: Fast and Secure Inter-Process Communication for Microkernels", EuroSys, April 2020
- **seL4 Performance**, https://sel4.systems/About/Performance/, accessed 2022-10-09

Functionality & Security Proofs

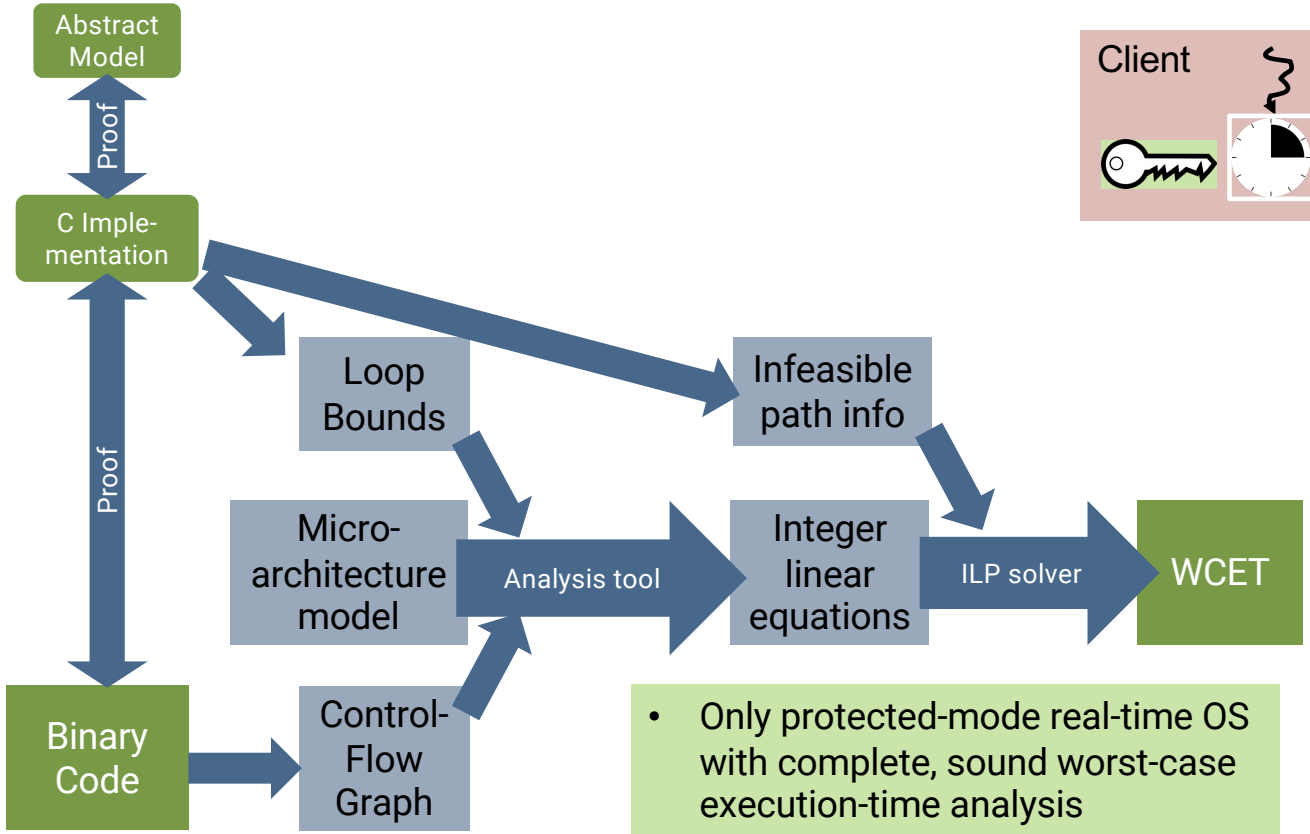


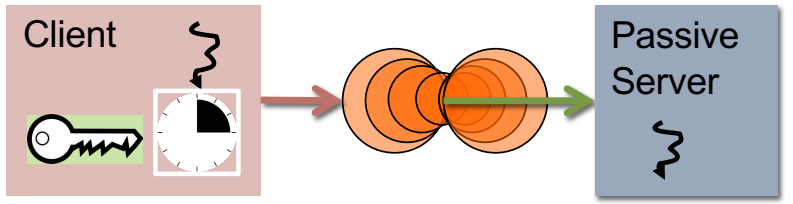
seL4: World's first OS kernel with correctness proof!

seL4: Still only capability-based OS kernel with correctness proof!

Details in June Andronick's talk

Unique Support for Protected Real Time



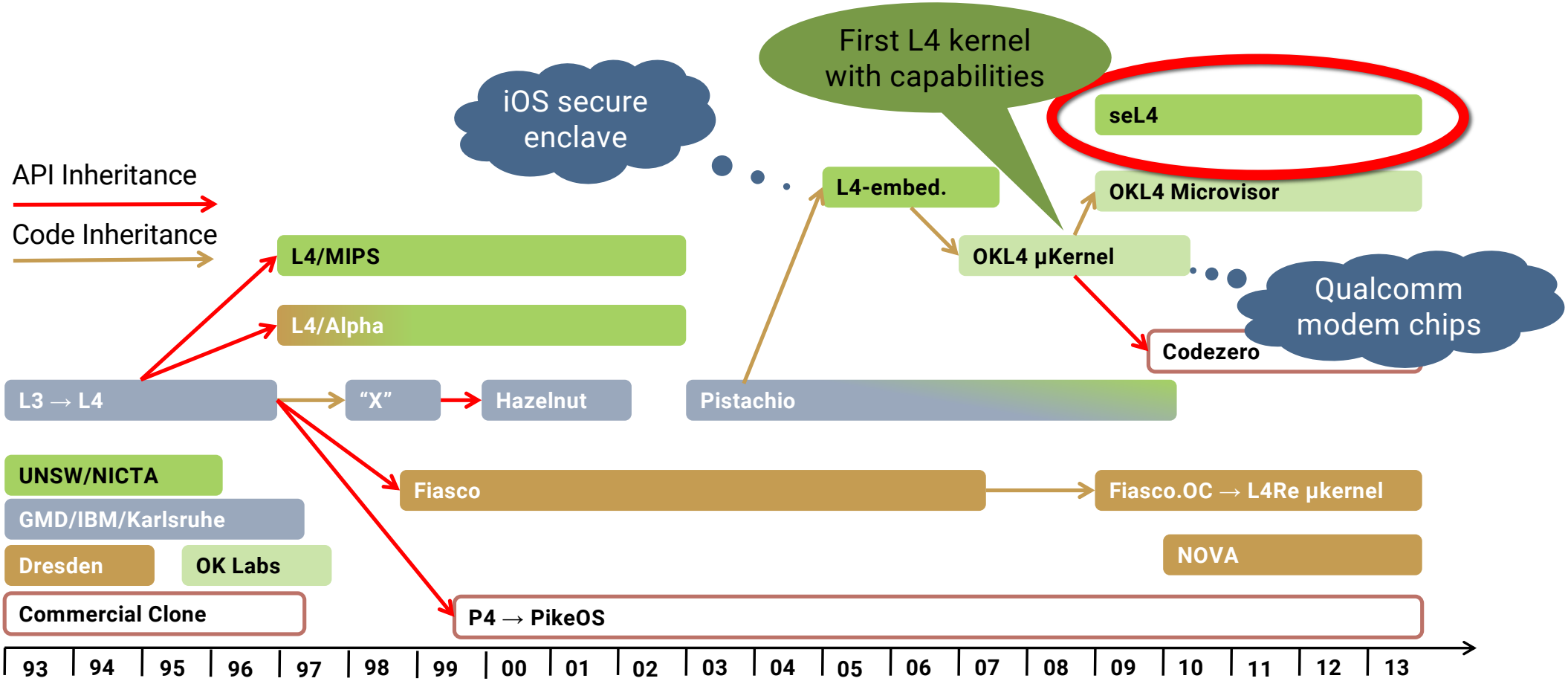


- Time as a first-class resource authorised by capabilities
- Prevent high-prio threads from dominating the CPU

- Only protected-mode real-time OS with complete, sound worst-case execution-time analysis
- High-assurance by connecting to correctness proofs

Note: Armv7 only
- insufficient timing info for modern processors
- Open RISC-V implementations should enable it again!

How Did We Get Here?



seL4's Philosophy & Principles

seL4 Principles

Proper microkernel:
- Minimal
- Provides policy-free mechanisms only
- Single access-control mechanism: Capabilities

Security:
- Suitable base for security-critical systems
- Provably correct and secure

Performance:
- Security is no excuse for poor performance!
- Don’t pay for what you don’t use

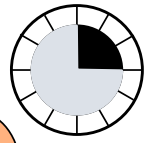
Anti-Principles:
- Hardware abstraction
- Prevent foot guns
- Usability

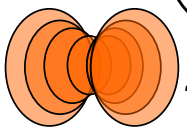
User-level issue!

The microkernel is the assembly language of operating systems!

Concepts in a Slide

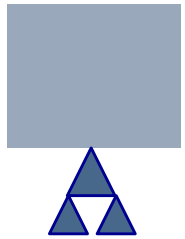
Capabilities (Caps): reference kernel objects

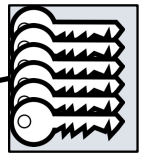


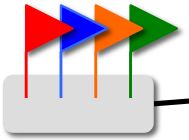


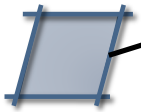
10 kernel object types:

- Threads (thread-control blocks: TCBs)
- Scheduling contexts (SCs)
- Address spaces (page table objects: PDs, PTs)
- Endpoints (IPC)
- Reply objects (ROs)
- Notifications
- Capability spaces (CNodes)
- Frames
- Interrupt objects (architecture specific)
- Untyped memory







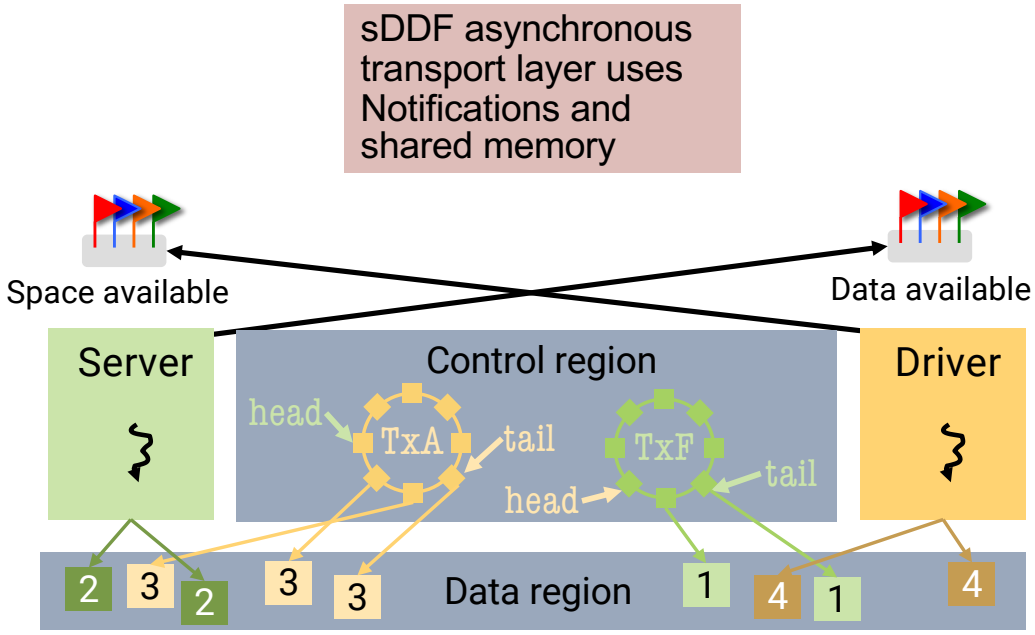


System calls:

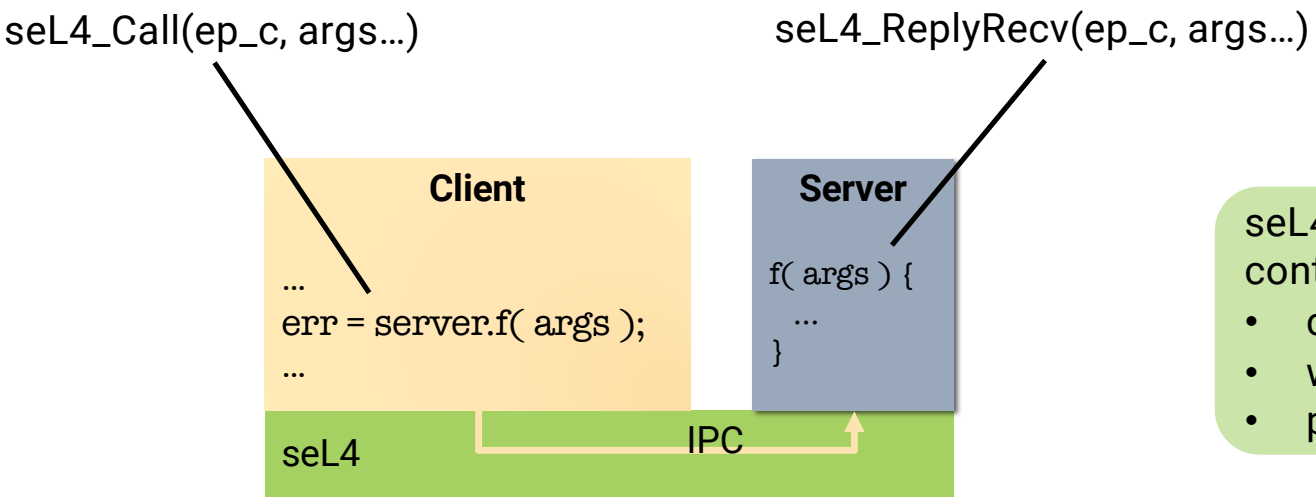
- Call(), ReplyRecv() (and one-way variants)
- Yield()

seL4 Usage

Are Endpoints a Minimality Violation?



Endpoints Are For Protected Procedure Calls!



Think: System call!

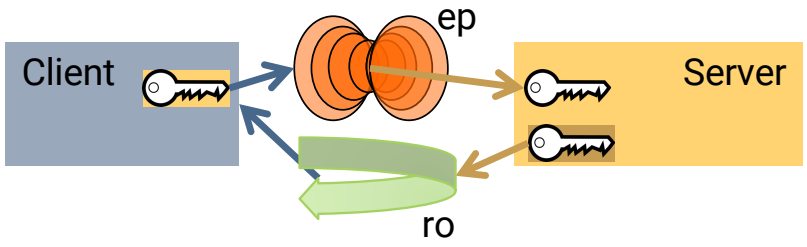
seL4 IPC **is**: A user-controlled context switch "with benefits":

- change protection context
- without scheduler invocation
- pass arguments / result

seL4 IPC is **not**:

- A mechanism for shipping data
- A synchronisation mechanism
  - side effect, not purpose

Protected Procedure Calls Done Right



Priorities:
- Call to high
- Receive from low!

Client | **Kernel** | **Server**

`Call(ep, args)` → *deliver to server*

block client on RO

`ReplyRecv(ro,ep,&args)`

One per client for blocking calls!

process

deliver to client ← `ReplyRecv(ro,ep,&args)`

process

Code Patterns

CLIENT

```
...
msg = Call (ep_c, op, arg...)
...
```

SERVER

```
...
msg=Recv(ep_c, &bdg, ro_c)
while (TRUE) {
  ...
  msg=ReplyRecv(ep_c, reply, &bdg, ro_c);
}
```

Protocol initialisation!

One-way operations only for
- **protocol initialisation**
- **exceptions**

Payload is for by-value syscall arguments, not bulk data

Endpoints are for Protected Procedure Calls

seL4 IPC **is**: A user-controlled context switch “with benefits”:

- change protection context
- without scheduler invocation
- pass arguments / result

Server runs on client’s SC

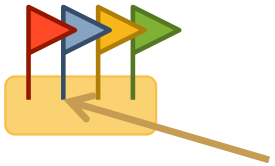
This makes no sense whatsoever across cores!!!!

Think: System call!

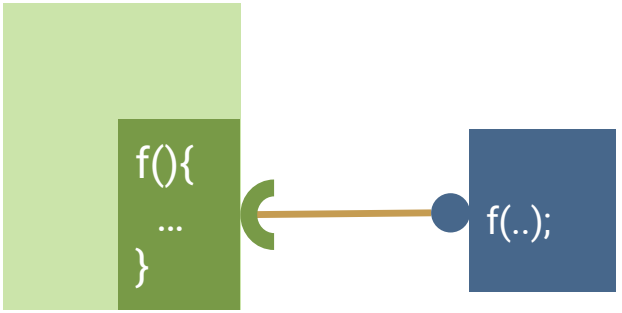
seL4 Core Platform Helps

Thin wrapper of seL4 abstractions but implying correct usage

Simple, event-driven programming model



Notification



Protected Procedure Call (PPC)

Synchronous

Communication Channel (CC)

Protection Domain (PD)

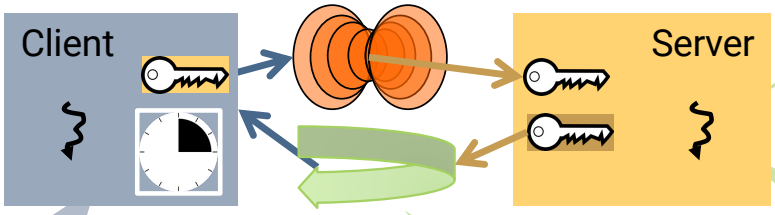
Protection Domain (PD)

Asynchronous

May be a VM

Memory Region (MR)

MCS: It's Not Just For Real Time ~~Anymore!~~



The MCS kernel provides essential mechanisms for ensuring timeliness in mixed-criticality systems

But on the way improves the seL4 model in several ways

Generally simplified server implementations

Passive servers complete protected procedure call model

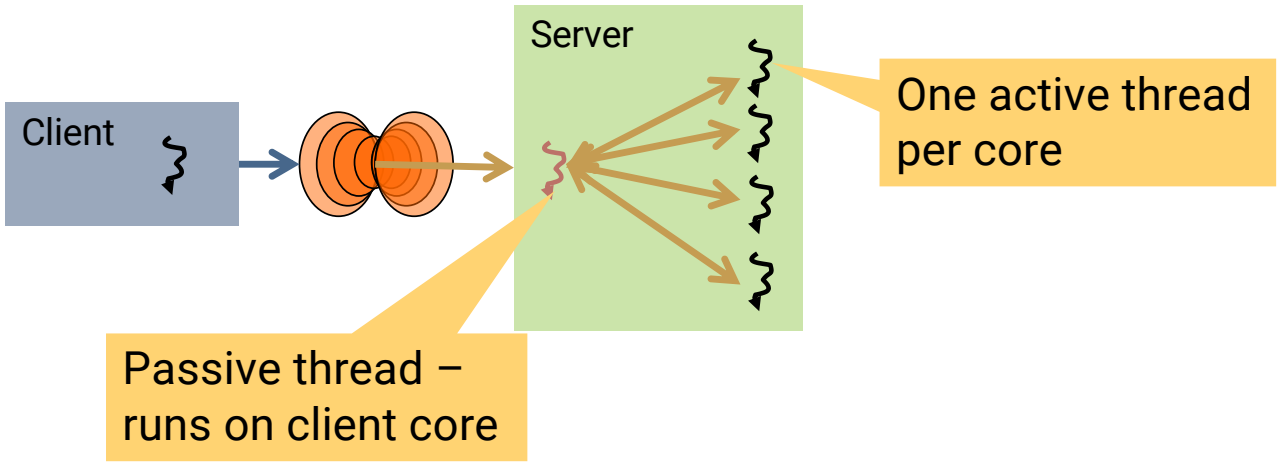
Leads to budgets, time as a first-class resource, capability-protected, principled time-slice donation

Reply objects simplify servers with blocking APIs

More combined system calls (used in sDDF)

How About A Multiprocessing Server?

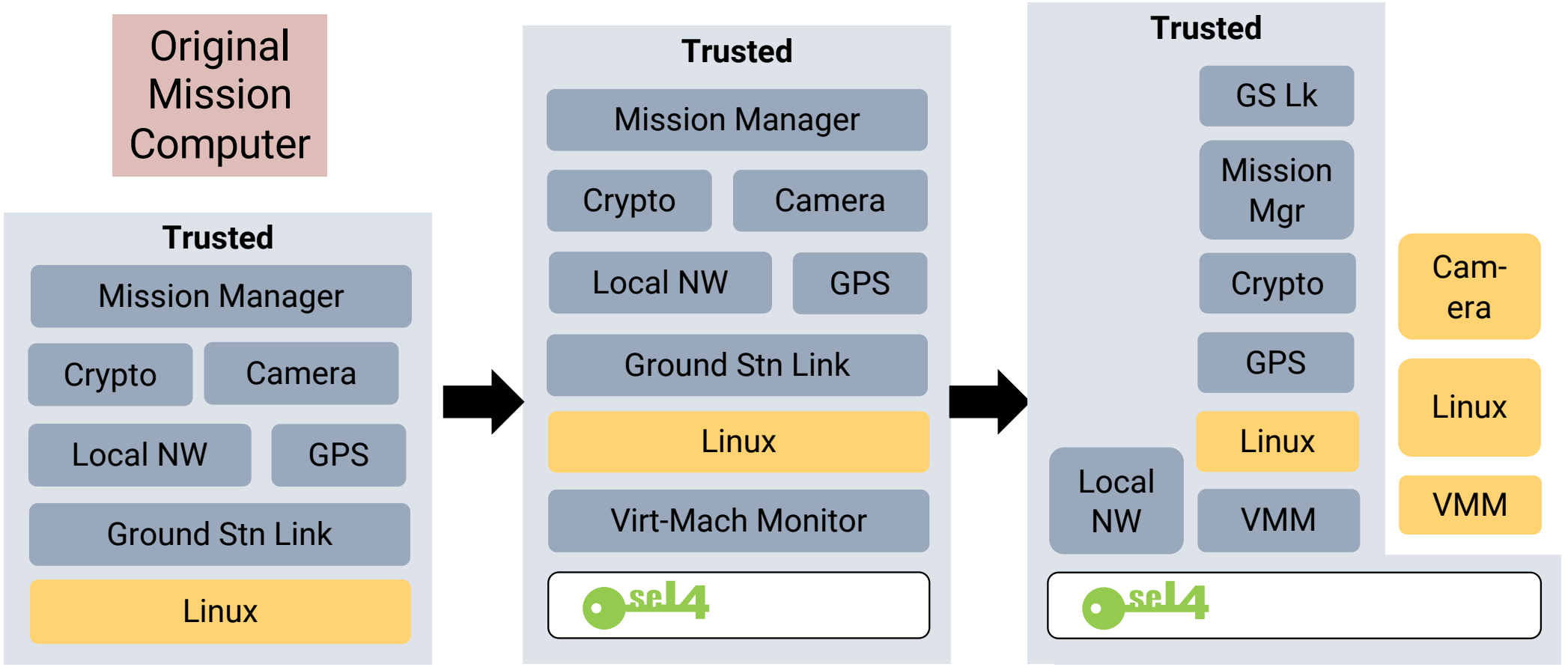
Multiprocessing is server policy, should be hidden from client!



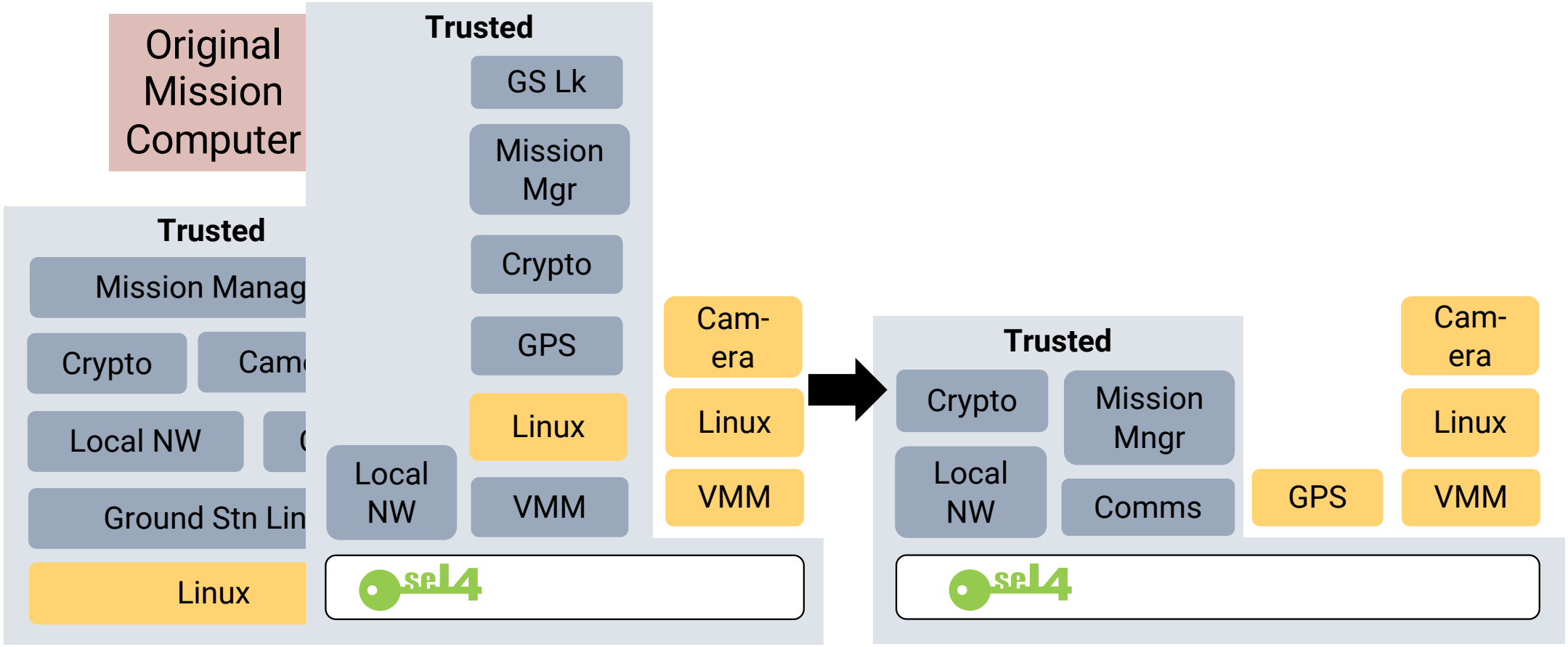
Note:
This only works cleanly with the MCS kernel!

Usage – Legacy Systems

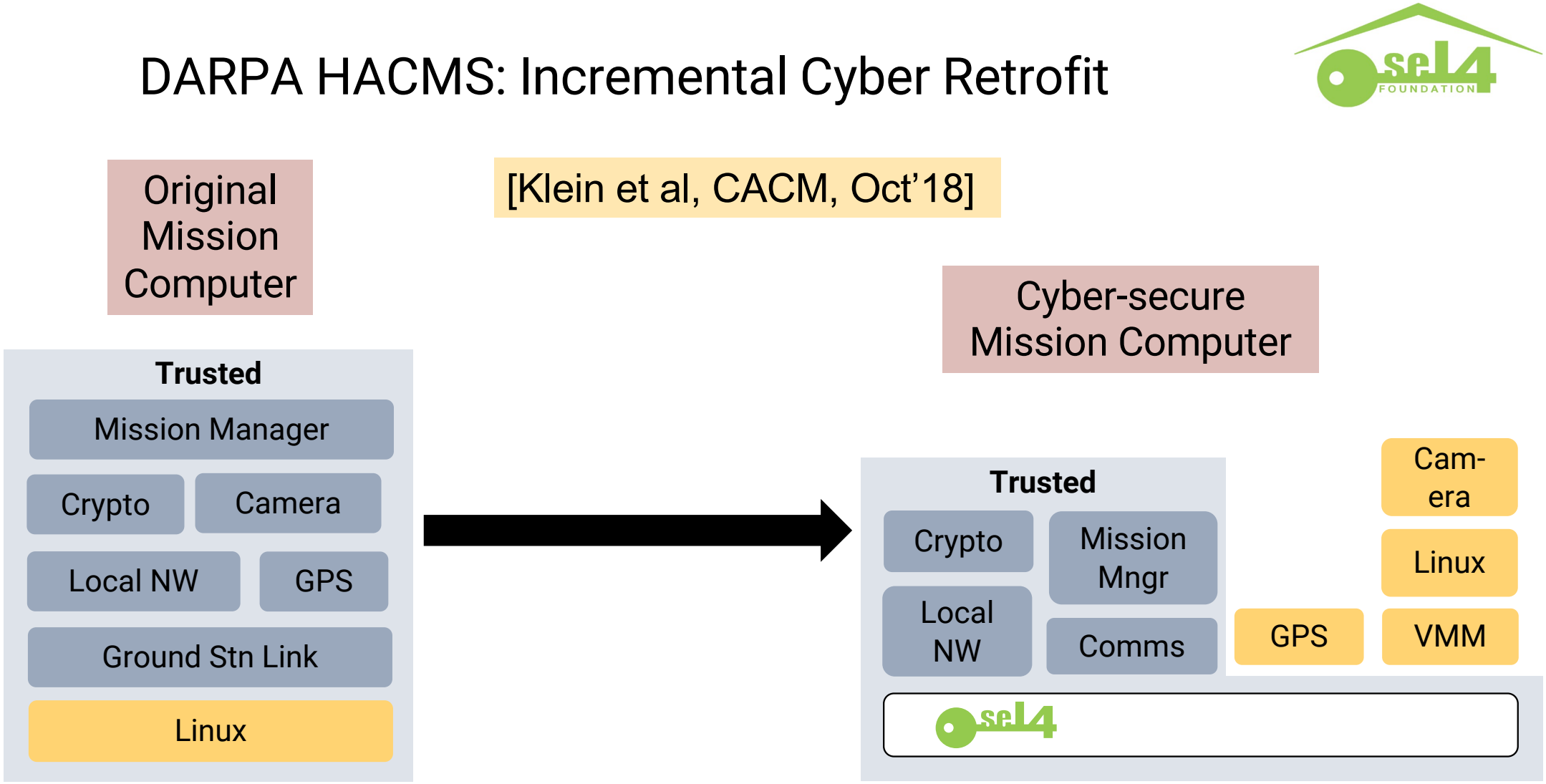
DARPA HACMS: Incremental Cyber Retrofit



DARPA HACMS: Incremental Cyber Retrofit



DARPA HACMS: Incremental Cyber Retrofit



World’s Most Secure Drone: DEFCON’21

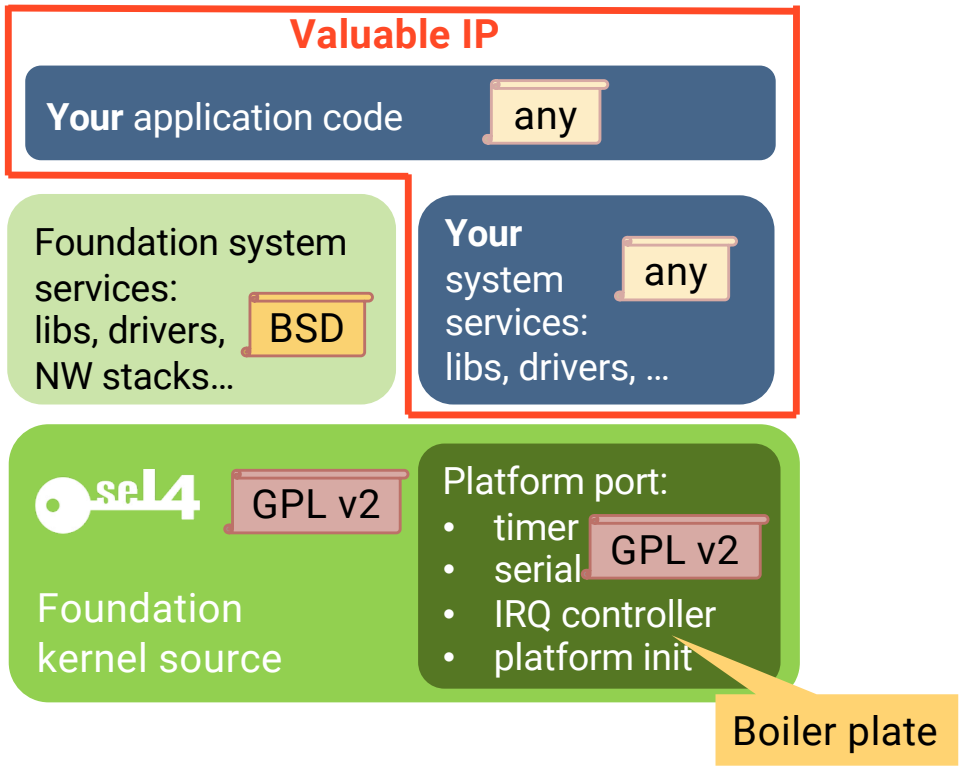
← Tweet

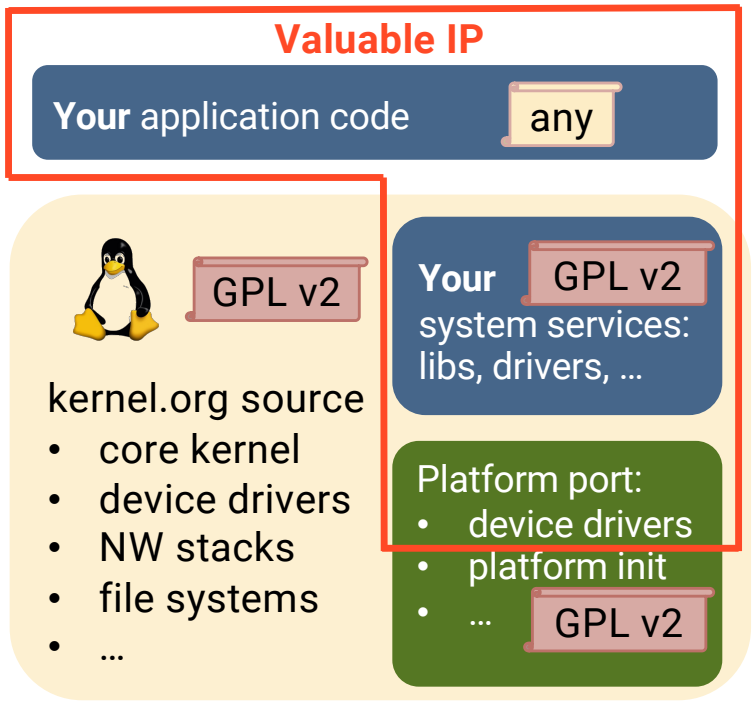
DARPA
@DARPA

We brought a hackable quadcopter with defenses built on our HACMS program to @defcon #AerospaceVillage.

Licensing

Licensing: What Does the GPL Imply?





Summary

- seL4 is unique and powerful
- To get the most out of it, you'll need to learn to use it correctly
- … or use the seL4 Core Platform

Defining the state of the art
in trustworthy operating systems
for 13 years – and counting!

Further Reading:

- More on seL4 principles: https://bit.ly/34uI8Fl
- seL4 whitepaper: https://sel4.systems/About/seL4-whitepaper.pdf