Reducing the reliance on verification experts for seL4 proofs

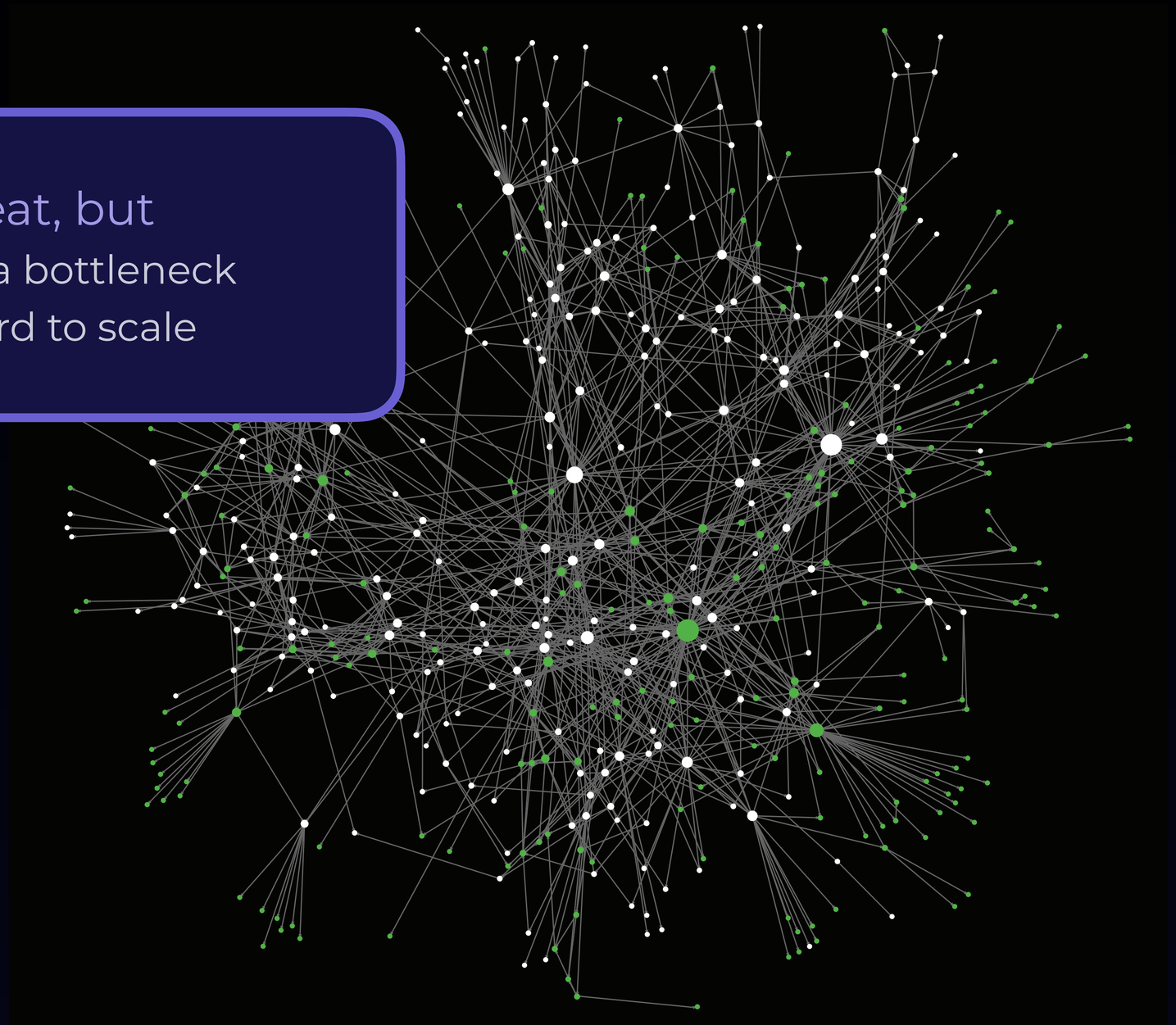
Gerwin Klein | seL4 summit 2023

faster + cheaper = better

sometimes you only need to pick two

How we plan to become faster and cheaper

- Formal verification is great, but
  - availability of experts still a bottleneck
  - development time still hard to scale



How we plan to become faster and cheaper

- Formal verification is great, but
  - availability of experts still a bottleneck
  - development time still hard to scale

- We can improve that
  - reduce proof maintenance cost
  - increase proof development speed
  - make specific verification tasks automatic

Automatic Verification of Platform Ports

Verified seL4 Architectures

- Architecture coverage fairly good
  - but proofs apply only to one platform/board for each architecture

Verified seL4 Architectures

- Architecture coverage fairly good
  - but proofs apply only to one platform/board for each architecture

- Platform coverage recently extended for Arm 32bit
  - 3 different verified Arm platforms: Sabre Lite, Exynos 5, IMX8MM-EVK
  - but git branches brittle in maintenance
  - not scalable to many boards

Verified seL4 Platforms (currently)

- https://docs.sel4.systems/Hardware/

  seL4 supports 32 different platforms

elect ARMv7 and ARMv8 Platforms.

RM Platforms

stem- -chip	Core	Arch	Virtualisation	IOMMU	Status
Exynos5	Cortex-A15	ARMv7A	ARM Hyp	No	Unverified
i.MX8MQ	Cortex-A53 Quad 1.5 GHz	ARMv8A	No	No	Unverified
OMAP3	Cortex-A8	ARMv7A	No	No	Unverified
AM335x	Cortex-A8	ARMv7A	No	No	Unverified
Kirin 620	Cortex-A53	ARMv8A	ARM HYP	No	Unverified
IMX8MM-	Cortex-A53	ARMv8A,	No	No	FC,

Arndale

Avnet MaaXBoard

BeagleBoard

BeagleBone Black / Blue

HiKey

Verified seL4 Platforms (currently)

- https://docs.sel4.systems/Hardware/

  seL4 supports 32 different platforms

6 of these support verification

19%

elect ARMv7 and ARMv8 Platforms.

RM Platforms

ystem-	Core	Arch	Virtualisation	IOMMU	Status
		ARMv7A	ARM Hyp	No	Unverified
		ARMv8A	No	No	Unverified
OMAP3	Cortex-A8	ARMv7A	No	No	Unverified
AM335x	Cortex-A8	ARMv7A	No	No	Unverified
Kirin 620	Cortex-A53	ARMv8A	ARM HYP	No	Unverified
IMX8MM-	Cortex-A53	ARMv8A,	No	No	FC,

BeagleBoard

BeagleBone Black / Blue

HiKey

Verified seL4 Platforms (plan)

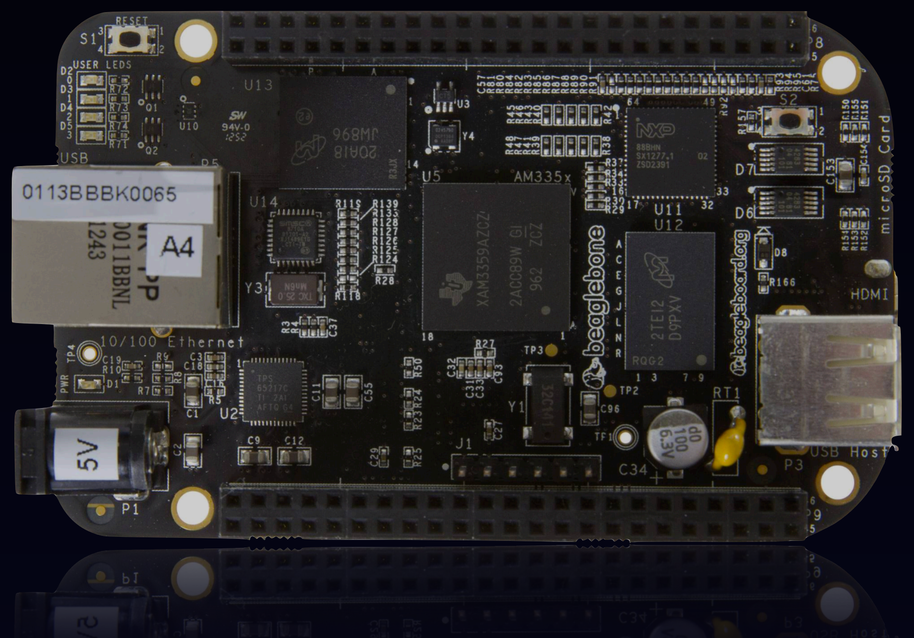
- Ideally, all platforms should have verification
- Even 90% would already be great

and ARMv8 Platforms.

ns

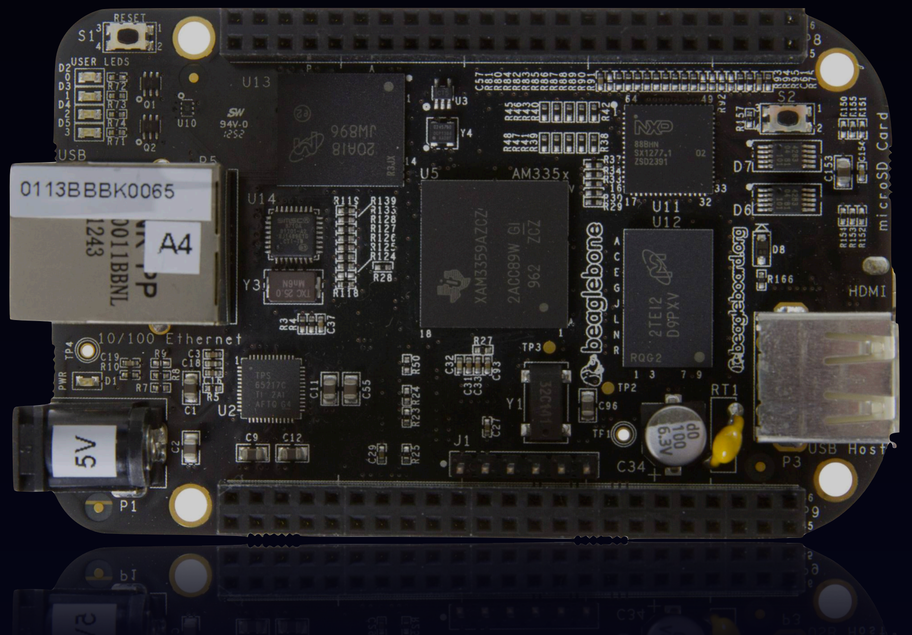
		Core	Arch	Virtualisation	IOMMU	Status
		Cortex-	ARMv7A	ARM Hyp	No	verified
			Mv8A	No	No	verified
			Mv7A	No	No	verified
BeagleBone Black / Blue	AM335x	Cortex-A8	ARMv7A	No	No	verified
HiKey	Kirin 620	Cortex-A53	ARMv8A	ARM HYP	No	verified
	IMX8MM-	Cortex-A53	ARMv8A,			verified

New seL4 Platforms (currently)



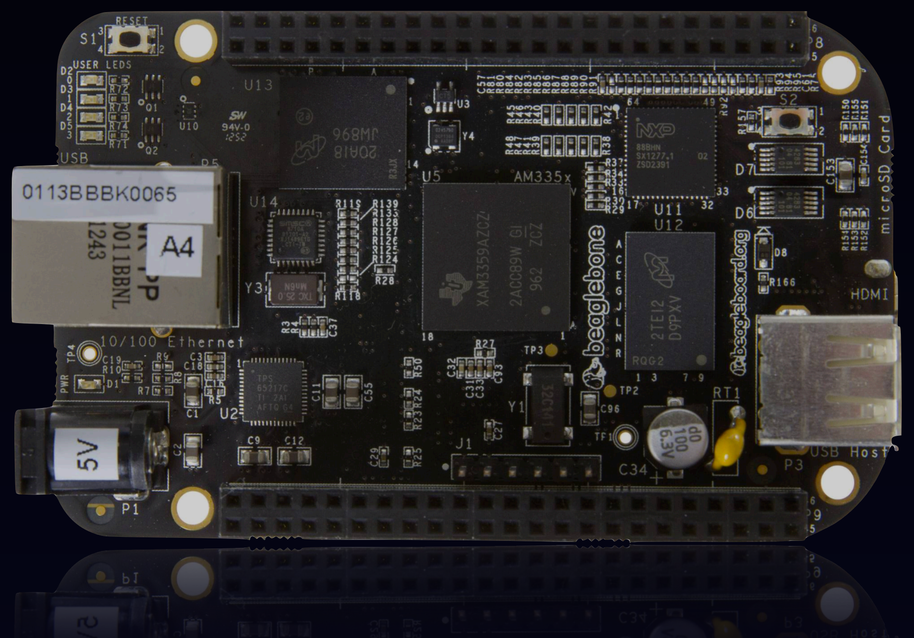
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅



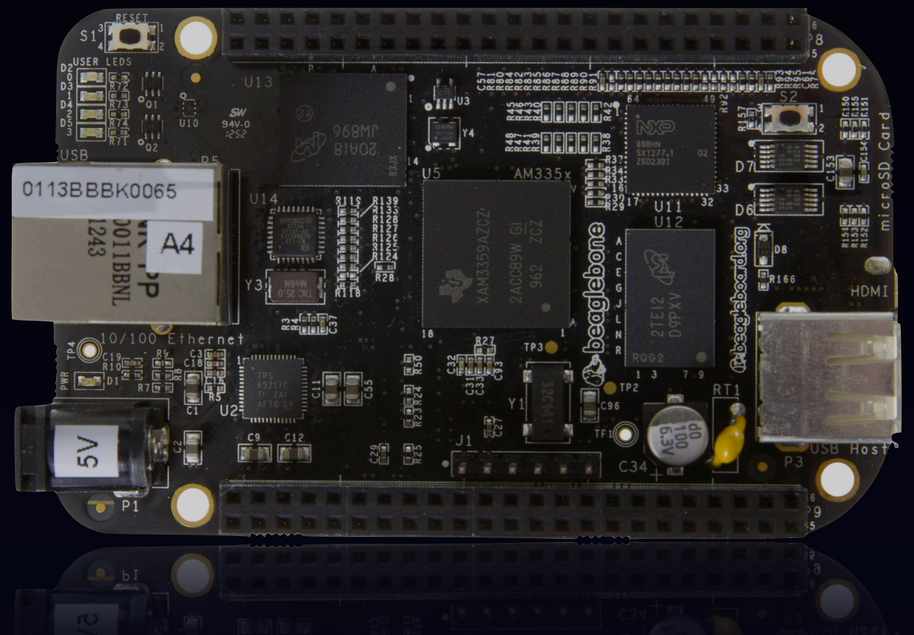
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update



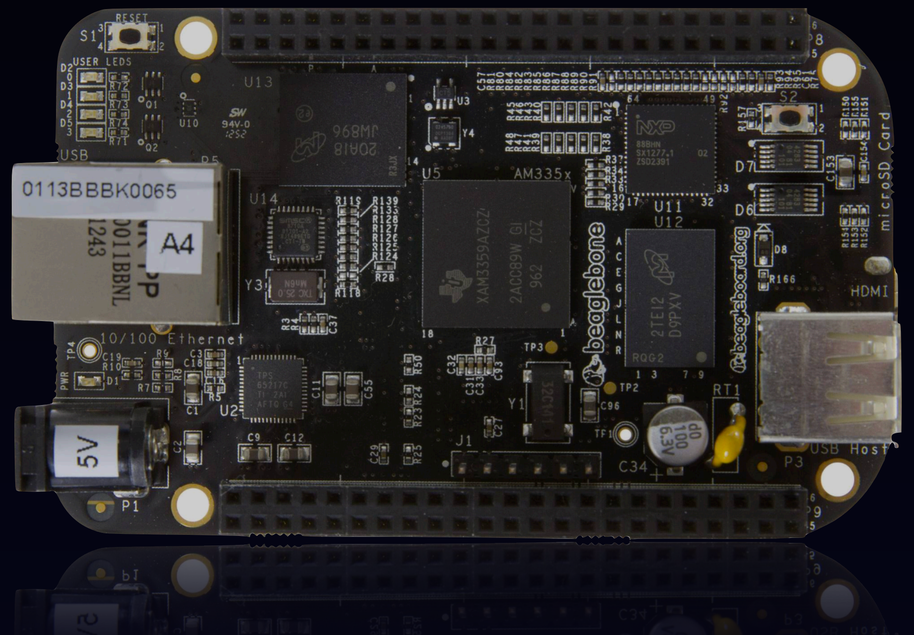
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅



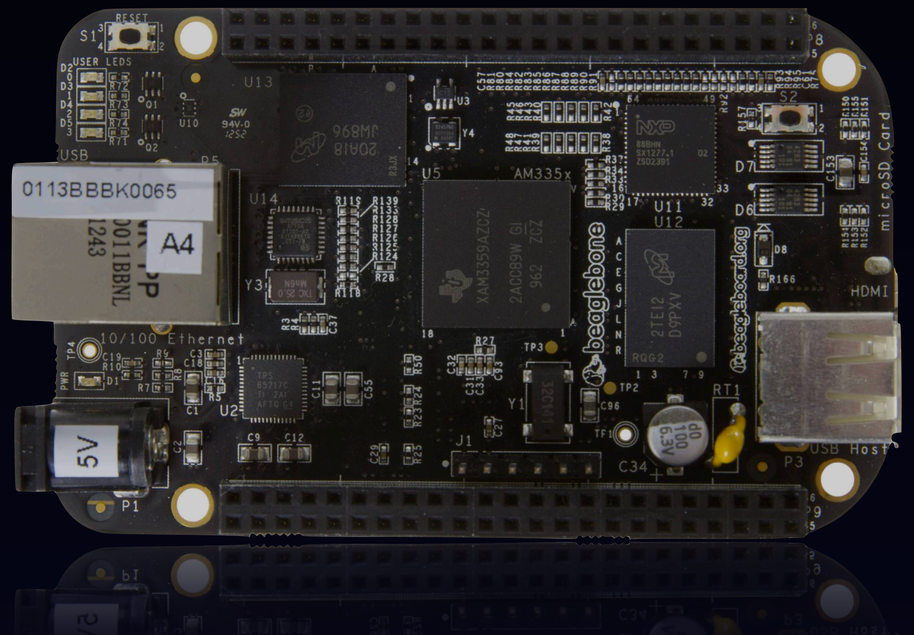
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅
- Effort between a few days and a few weeks. 👍



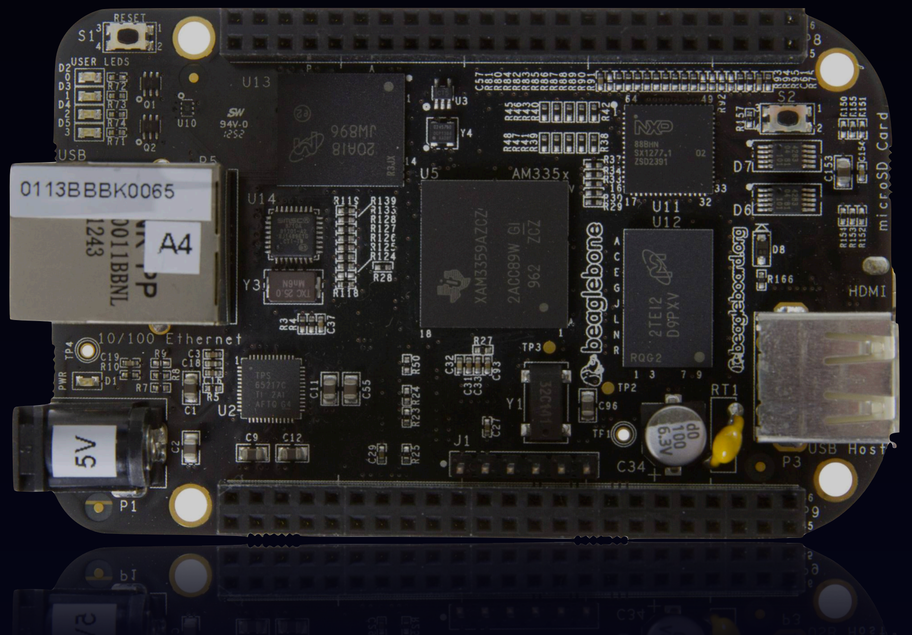
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅
- Effort between a few days and a few weeks. 👍
- We perform the proof update, add a branch to the proofs. 😎



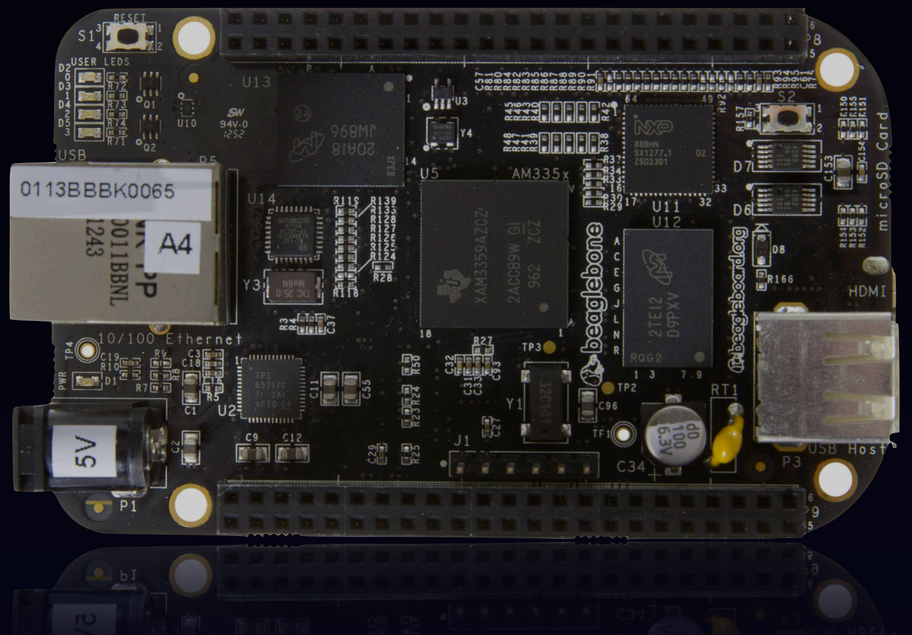
New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅
- Effort between a few days and a few weeks. 👍
- We perform the proof update, add a branch to the proofs. 😎
- Proof test combinations multiply 😕



New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅
- Effort between a few days and a few weeks. 👍
- We perform the proof update, add a branch to the proofs. 😎
- Proof test combinations multiply 😕
- Maintenance cost increases 😕



New seL4 Platforms (currently)

- Follow kernel porting instructions ✅
- Now need a proof update
- Can contract Proofcraft ✅
- Effort between a few
- We perform the proo
- Proof test combinati
- Maintenance cost in

- Update usually easy
- Maintaining multiple proof versions is painful
  (> 6h run time for a single proof check)
- When something does break, many copies to update
- Makes maintenance boring and expensive
- High incentive to eliminate these completely

New seL4 Platforms (plan)

New seL4 Platforms (plan)

- Follow porting instructions ✅

New seL4 Platforms (plan)

- Follow porting instructions ✅
- Build automatically checks conditions and generates proof input ✅

New seL4 Platforms (plan)

- Follow porting instructions ✅
- Build automatically checks conditions and generates proof input ✅
- If you want to be really sure: re-run proof ✅

New seL4 Platforms (plan)

- Follow porting instructions ✅
- Build automatically checks conditions and generates proof input ✅
- If you want to be really sure: re-run proof ✅
- The end 🎉

New seL4 Platforms (plan)

- Follow porting instructions ✅
- Build automatically checks conditions and generates proof input ✅
- If you want to be really sure: re-run proof ✅
- The end 🎉

- No need for verification experts in most cases
- No new branches or maintenance explosion
- When build checks fail, there is likely a real problem that needs deeper expertise

Why and How?

What changes in a platform port

- Platform parameters
- Config parameters

```
/ {
        chosen {
                seL4,elfloader-devices =
                    "serial0";
                seL4,kernel-devices =
                    "serial0",
                    &{/ocp/interrupt-controller@48200000},
                    /* The following devices are used to s
                    /* dmtimer4, OMAP Dual-Mode timer */
                    &{/ocp/timer@48044000},
                    &{/ocp/l4_wkup@44c00000/prcm@200000},
                    &{/ocp/wdt@44e35000}; /* Watchdog time
        };
};
```

```
declare_platform(am335x KernelPlatformAM335X PLAT_AM335X KernelSel4ArchAarch32)
set(c_configs PLAT_AM335X_BONEBLACK PLAT_AM335X_BONEBLUE PLAT_AM335X_BONE)
set(
    cmake_configs
    KernelPlatformAM335XBoneBlack
    KernelPlatformAM335XBoneBlue
    KernelPlatformAM335XBone
)
set(plat_lists am335x-boneblack am335x-boneblue am335x-bone)
foreach(config IN LISTS cmake_configs)
    unset(${config} CACHE)
endforeach()

if(KernelPlatformAM335X)
    declare_seL4_arch(aarch32)

    set(KernelHardwareDebugAPIUnsupported ON CACHE INTERNAL "")

    set(KernelArmCortexA8 ON)
    set(KernelArchArmV7a ON)
    check_platform_and_fallback_to_default(KernelARMPlatform "am335x-boneblack")
    list(FIND plat_lists ${KernelARMPlatform} index)
    if("${index}" STREQUAL "-1")
        message(FATAL_ERROR "Which am335x platform not specified")
    endif()
```

What changes in a platform port

- Platform parameters
- Config parameters

- Platform parameters:
  - memory regions
  - devices addresses
  - board features (FPU, IRQ controller, HYP, SMMU, etc)

```
declare_platform(am335x KernelPlatformAM335X PLAT_AM335X KernelSel4ArchAarch32)
set(c_configs PLAT_AM335X_BONEBLACK PLAT_AM335X_BONEBLUE PLAT_AM335X_BONE)
set(
    cmake_configs
    KernelPlatformAM335XBoneBlack
    KernelPlatformAM335XBoneBlue
```

```
oneblue am335x-bone)
```

```
d ON CACHE INTERNAL "")
```

```
set(KernelArmCortexA8 ON)
set(KernelArchArmV7a ON)
check_platform_and_fallback_to_default(KernelARMPlatform "am335x-boneblack")
list(FIND plat_lists ${KernelARMPlatform} index)
if("${index}" STREQUAL "-1")
    message(FATAL_ERROR "Which am335x platform not specified")
endif()
```

```
                &{/ocp/wdt@44e35000}; /* Watchdog time
        };
};
```

What changes in a platform port

- Platform parameters
- Config parameters

- Platform parameters:
  - memory regions
  - devices addresses
  - board features (FPU, IRQ controller, HYP,

- Config parameters:
  - MCS on/off
  - fast path on/off
  - number of domains
  - max retype fan-out
  - init CNode size
  - etc

```
declare_platform(am335x KernelPlatformAM335X PLAT_AM335X KernelSel4ArchAarch32)
set(c_configs PLAT_AM335X_BONEBLACK PLAT_AM335X_BONEBLUE PLAT_AM335X_BONE)
set(
    cmake_config
    KernelPlatf
    KernelPlatf
```

```
                &{/ocp/wdt@44e35000}; /* Watchdog time
        };
};
```

```
set(KernelA
set(KernelA
check_platf                                              x-boneblack")
list(FIND pla
if("${index}" STREQUAL "-1")
    message(FATAL_ERROR "Which am335x platform not specified")
endif()
```

What changes in a platform port

- Platform parameter
- Config parameters

Can be automated

- Platform parameters:
  - memory regions
  - devices addresses
  - board features (FPU, IRQ controller, HYP,

- Config parameters:
  - MCS on/off
  - fast path on/off
  - number of domains
  - max retype fan-out
  - init CNode size
  - etc

What changes in a platform port

- Platform parameters
- Config parameters

```
declare_platform(am335x KernelPlatformAM335X PLAT_AM335X KernelSel4ArchAarch32)
set(c_configs PLAT_AM335X_BONEBLACK PLAT_AM335X_BONEBLUE PLAT_AM335X_BONE)
set(
    cmake_config
    KernelPlatf
    KernelPlatf
```

- Platform parameters:
  - memory regions
  - devices addresses
  - board features (FPU, IRQ controller, HYP,

- Config parameters:
  - MCS on/off
  - fast path on/off
  - number of domains
  - max retype fan-out
  - init CNode size
  - etc

selects different code,
will be harder

```
    set(KernelA
    set(KernelA
    check_platf                                          x-boneblack")
    list(FIND pla
    if("${index}" STREQUAL "-1")
        message(FATAL_ERROR "Which am335x platform not specified")
    endif()
```

What changes in a platform port

- Platform parameters
- Config parameters

```
declare_platform(am335x KernelPlatformAM335X PLAT_AM335X KernelSel4ArchAarch32)
set(c_configs PLAT_AM335X_BONEBLACK PLAT_AM335X_BONEBLUE PLAT_AM335X_BONE)
set(
    cmake_config
    KernelPlatf
    KernelPlatf
```

- Platform parameters:
  - memory regions
  - devices addresses
  - board features (FPU, IRQ controller, HYP,

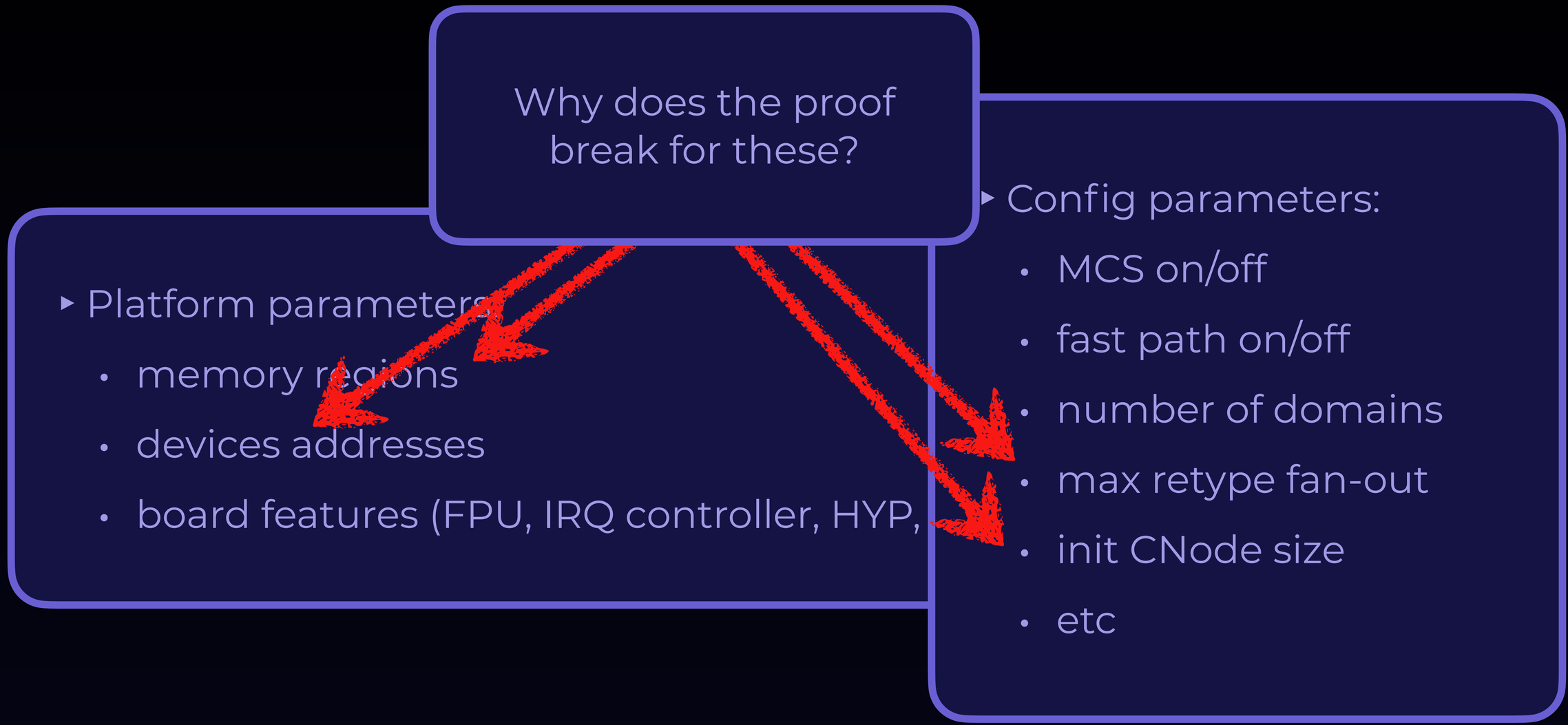
- Config parameters:
  - MCS on/off
  - fastpath on/off
  - number of domains
  - max retype fan-out
  - init CNode size

selects different proof setup

```
        &{/ocp/wdt@44e35000}; /* Watchdog time
    };
};
```

```
set(Ker
set(Ker
check_p
list(FI
if("${i
    mes
endif()
```

Why is this even an issue?



Why is this even an issue?

Why is this even an issue?

- Current platform and config constants:
  - fixed values in the spec
  - tied to a specific board, validated against that board
  - tied to specific config

Why is this even an issue?

- Current platform and config constants:
  - fixed values in the spec
  - tied to a specific board, validated against that board
  - tied to specific config

- Plan:
  - generate spec + config values from C config
  - validation still needs to happen manually!
  - this kind of validation usually easier with concrete testing
  - proof could be changed such that it happens to work with all reasonable values

Why is this even an issue?

- Current platform and config constants:
  - fixed values in the spec
  - tied to a specific board, validated against that board
  - tied to specific config

- Plan:
  - generate spec + config values from C config
  - validation still needs to happen manually!
  - this kind of validation usually easier with concrete testing
  - proof could be changed such that it happens to work with all reasonable values

- Not sufficient:
  - still leads to combination explosion
  - still needs you to re-run the full proofs for each change

Why is this even an issue?

- Current platform and config constants:
  - fixed values in the spec
  - tied to a specific board, validated against that board
  - tied to specific config
- Plan:
  - generate spec + confi
  - validation still needs t
  - this kind of validation
  - proof could be change
- Not sufficient:
  - still leads to combinat
  - still needs you to re-ru
- Instead: proof parameterisation
  - find sufficient conditions, e.g.:
    - physical base address must be aligned to x bits
    - must be greater than y
    - must be smaller than other config value
  - prove once:
    - all values that satisfy these conditions are safe

What about FPU, GIC etc

One single proof that works with and without FPU?

What about FPU, GIC etc

One single proof that works with and without FPU?

Raf says: don’t promise that to people yet.

What about FPU, GIC etc

One single proof that works with and without FPU?

Raf says: don’t promise that to people yet.

Ok, I won’t, but I have a bunch of ideas, and they might work.

The goal

- Follow porting instructions
- Build automatically checks conditions and generates proof input
- If you want to be really sure: re-run proof
- The end

The goal

- Follow porting instructions
- Build automatically checks conditions and generates proof input
- If you want to be really sure: re-run proof
- The end

- Looks achievable for a large set of platforms and config changes
- We'll be working on it.

When can I have this?

Not the only thing we’re working on..

Not the only thing we’re working on..

- Current verification projects:

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024
  - MCS verification: ongoing for RISC-V, planned for AArch64

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024
  - MCS verification: ongoing for RISC-V, planned for AArch64
  - Multikernel verification: ongoing

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024
  - MCS verification: ongoing for RISC-V, planned for AArch64
  - Multikernel verification: ongoing
  - **Agile proofs:** funding likely, starting 2024

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024
  - MCS verification: ongoing for RISC-V, planned for AArch64
  - Multikernel verification: ongoing
  - **Agile proofs:** funding likely, starting 2024
    - **platform proof automation**

Not the only thing we're working on..

- Current verification projects:
  - AArch64 verification: functional correctness in Mar 2024
  - MCS verification: ongoing for RISC-V, planned for AArch64
  - Multikernel verification: ongoing
  - **Agile proofs:** funding likely, starting 2024
    - **platform proof automation**
    - more generic proofs, more agility

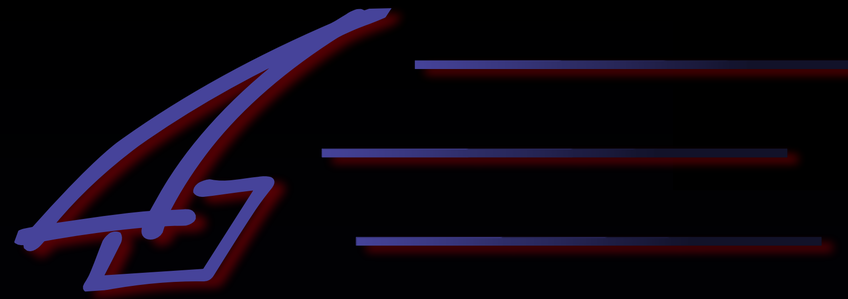
Summary

Towards more agile proofs

The Plan:

- Platform ports: no experts required
- Proof maintenance: faster and cheaper
- Proof engineers: happier and less of a bottleneck

Towards more agile proofs



The Plan:

- Platform ports: no experts required
- Proof maintenance: faster and cheaper
- Proof engineers: happier and less of a bottleneck

More capacity for features and updates

Thank You