CHERI Macaroons: Efficient, host-based access control for Cyber Physical Systems (CPS)

Michael Dodson, Alastair R. Beresford, Alexander Richardson,
Jessica Clarke, Robert N. M. Watson

Definitions

Cyber Physical Systems (CPS) tightly couple hardware and software with sensing and manipulation of the physical environment

- Automotive, industrial, robotics, medical

A **capability** is a token that confers the right to access an object

- CHERI capabilities
- Macaroon network tokens

Outline

- CPS security challenges
- Capability-based access control design pattern
  - CHERI capabilities for local access control
  - Macaroon tokens for network access control
- Concrete implementations and initial performance assessments

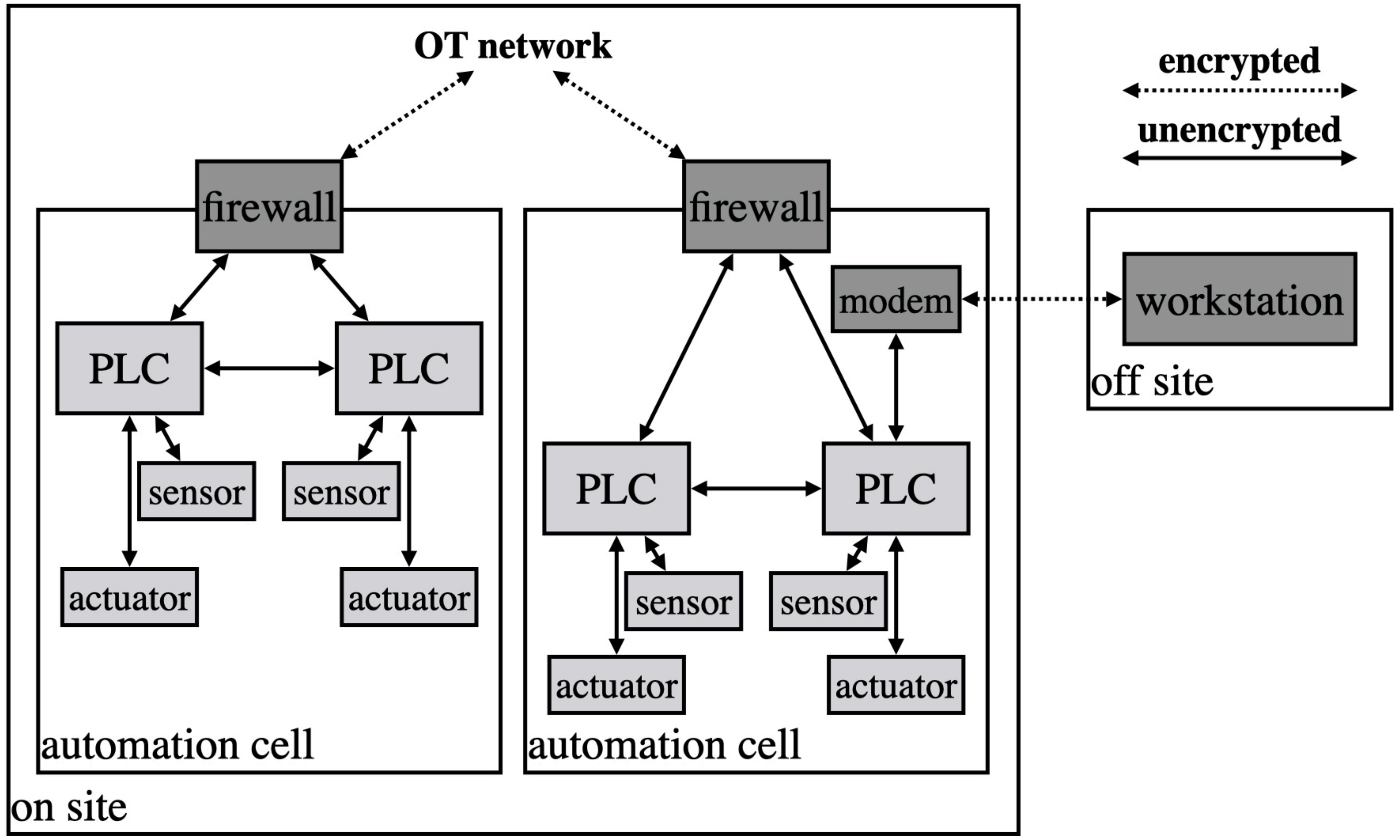
CPS security challenges

Domains:

- Industrial control
- Automotive
- Robotics
- Medical

Unique considerations:

- Decades-long lifetimes
- Remote deployments
- Piecemeal replacement
- Safety-critical functions
- Expensive certification



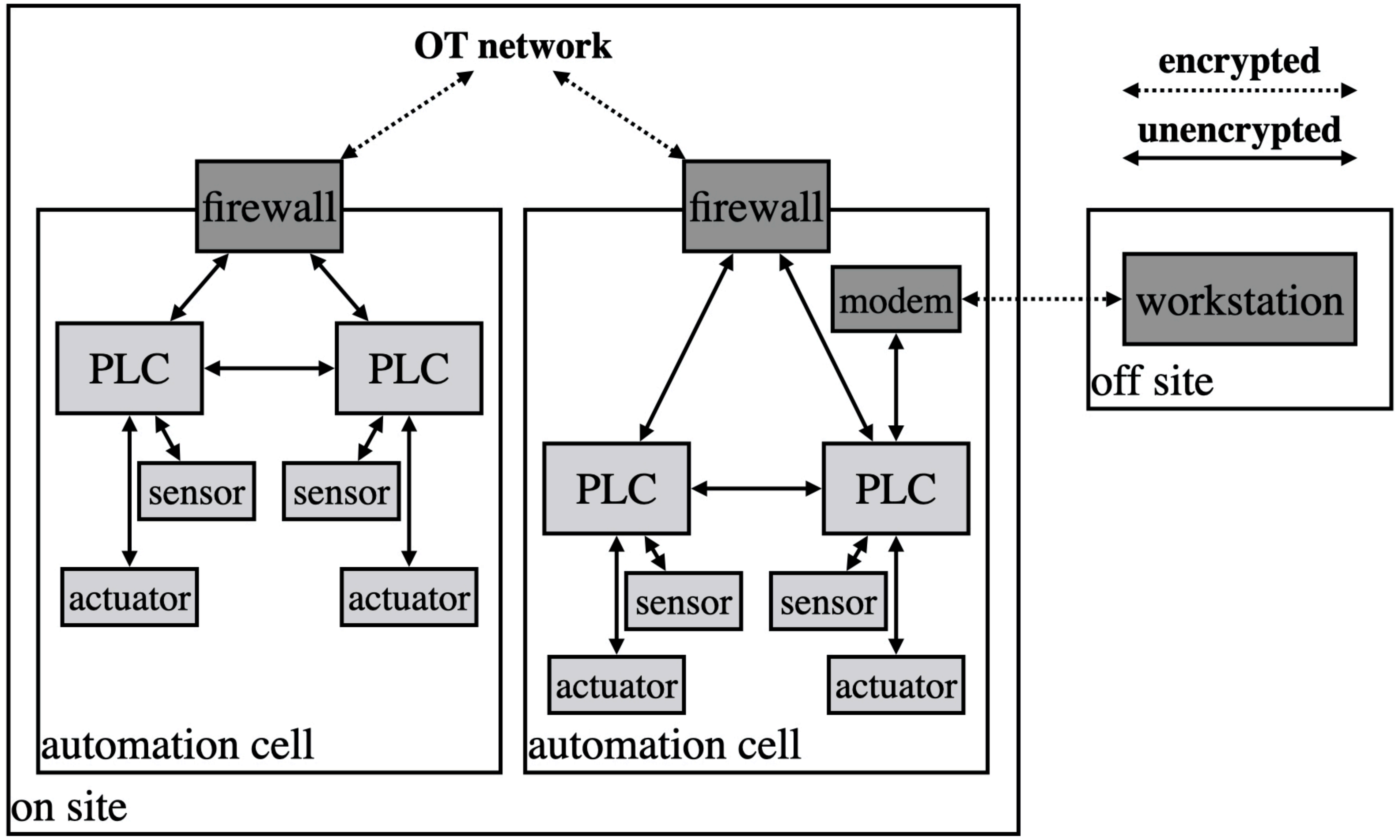
CPS security challenges

Challenges

- Legacy protocols
- Remote monitoring and maintenance
- Limited compute and memory resources
- Heterogeneous device networks
- Flat memory spaces

Current solutions

- Boundary protection
- Intrusion/anomaly detection



Capability-based distributed system

Proposal: Capabilities provide an intuitive and efficient mechanism for controlling access to physical 'objects' both on-device and between devices.

We introduce:

- Hardware-backed capabilities as tokens to protect access to physical resources
- A model for implementing hardware-backed tokens in distributed systems
- Efficient translation between hardware and network tokens

Benefits:

- Decouple authentication and authorization to offload non-real time tasks
- Integrity protection for insecure, legacy protocols
- Natural support for static, device-to-device communication graphs

Capability primer: Capability types

Architectural capabilities (local)

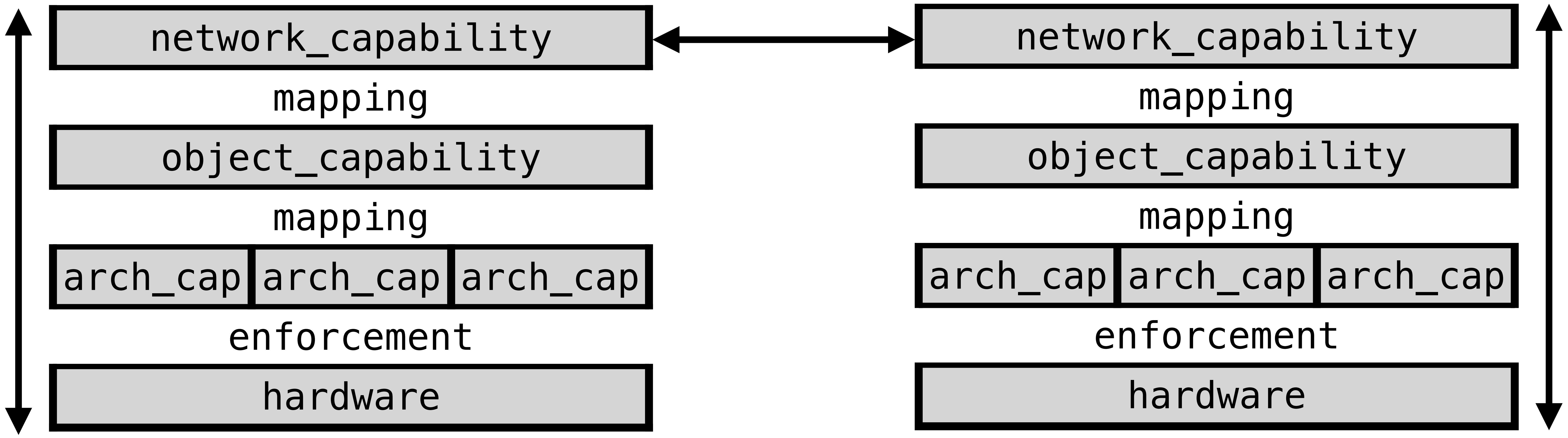
- Hardware defined
- Operations on memory

Object capabilities (local)

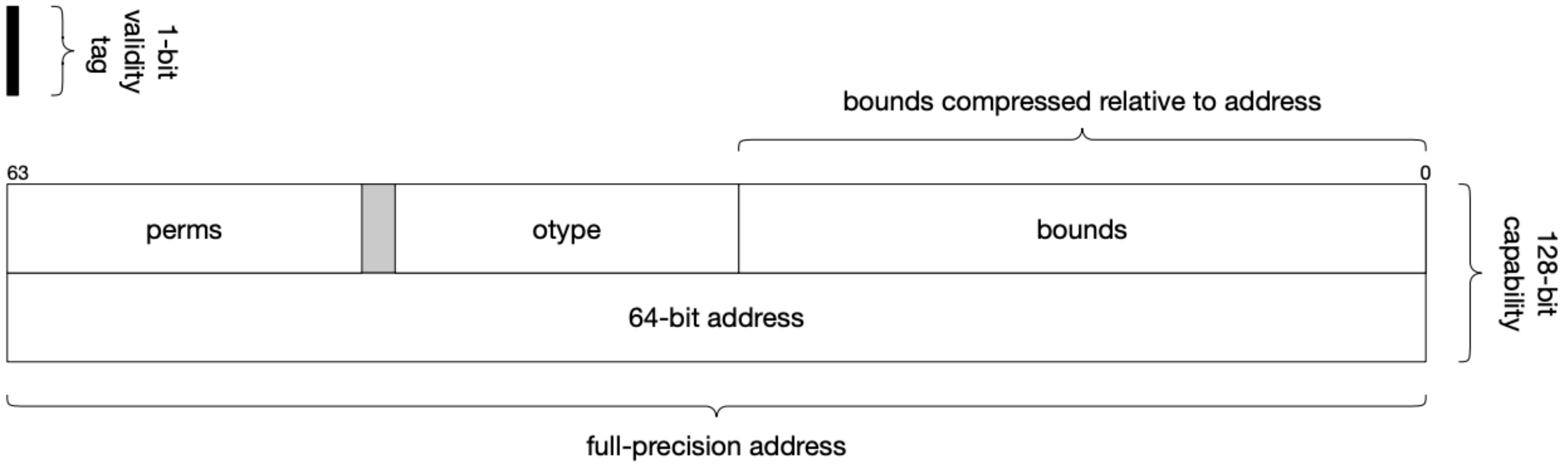
- Software defined
- Operations on objects

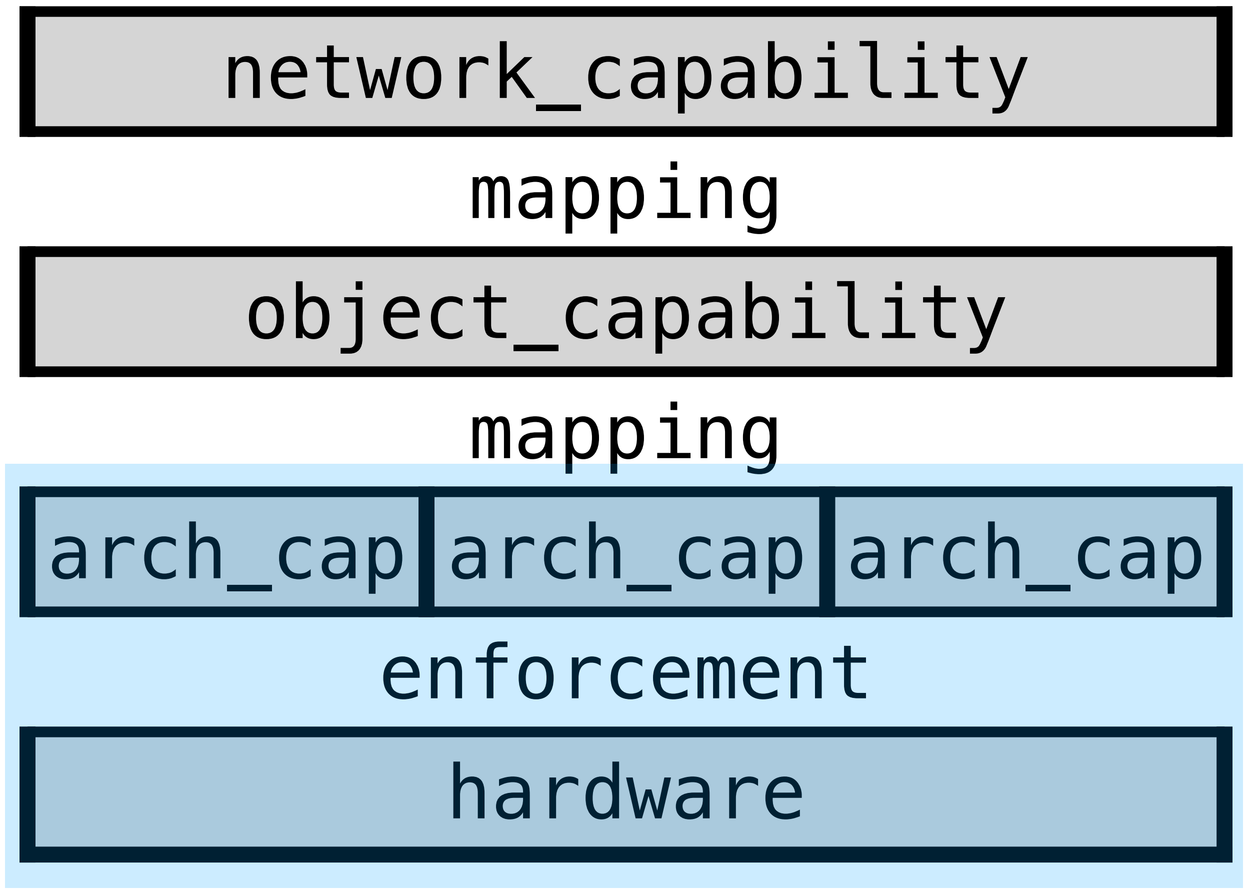
Network capabilities (distributed)

- Software defined
- Operations on objects
- Network instantiation of a local, object capability



CHERI architecture: Pointers become capabilities

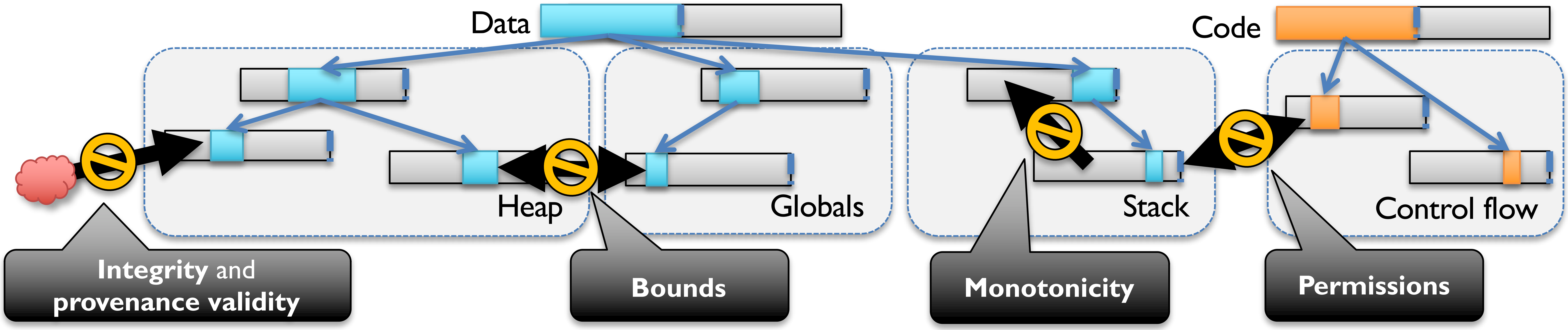




CHERI: Capability Hardware Enhanced RISC Instructions (Watson et al. CHERI ISAv7, 2018)

- Architecturally-defined “fat pointer” with OOB validity bit used to access a memory “object”
- Carries a base, length, offset, and permissions to limit memory access range and type
- CPU instructions govern legal operations on capabilities (e.g., maintain provenance, integrity, and monotonicity)
- CPU enforces bounds and permissions on dereference operations
- omposes with “host” ISA: MIPS, RISC-V, Arm Morello

CHERI architecture: Pointers become capabilities



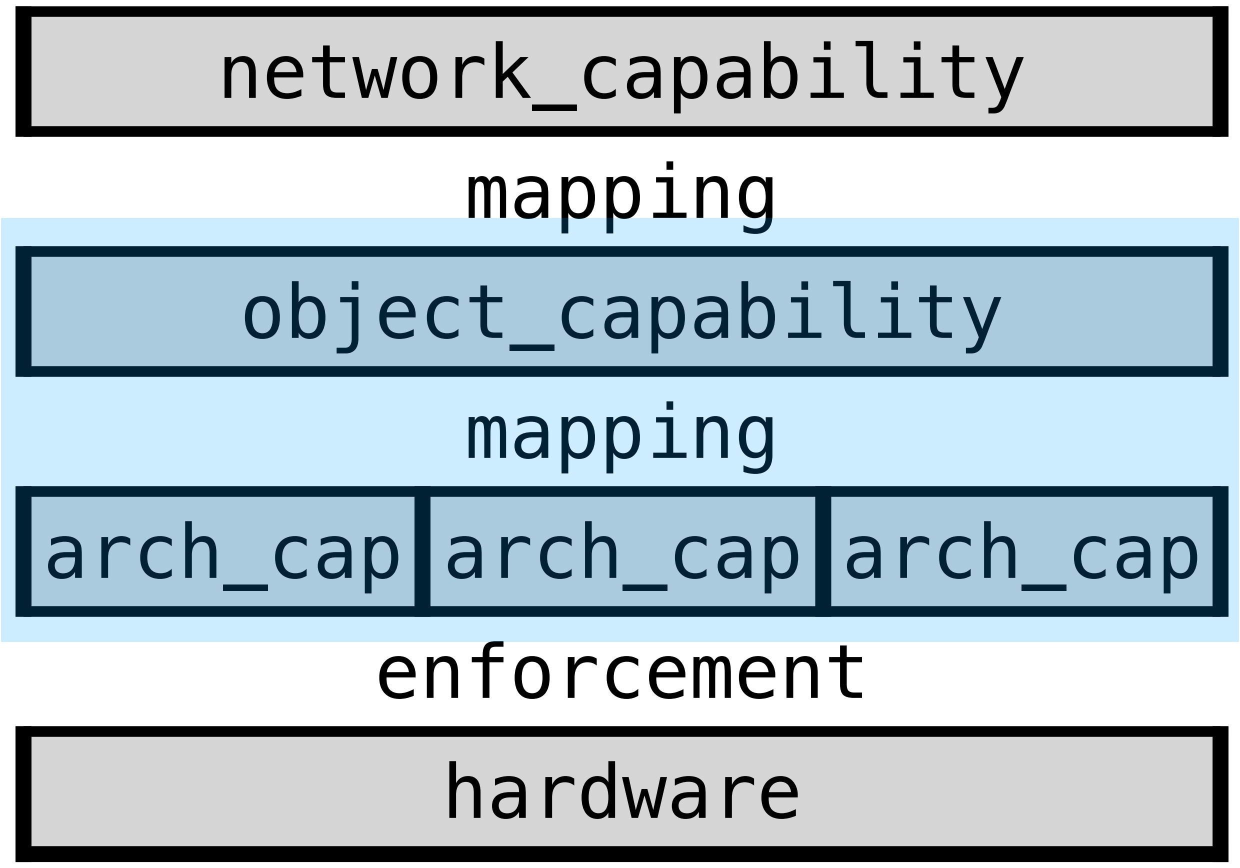
CHERI protects against:

- Creating valid pointers from integer data
- Accessing globals from the heap
- Dereferencing memory from a parent capability
- Executing a capability for a data object

Other benefits for CPS:

- Software-defined isolation without MMUs
- Temporal memory safety
- High compatibility with existing code

CHERI-based object capabilities



Token conferring access to software-defined 'objects'

- E.g., sensors, actuators

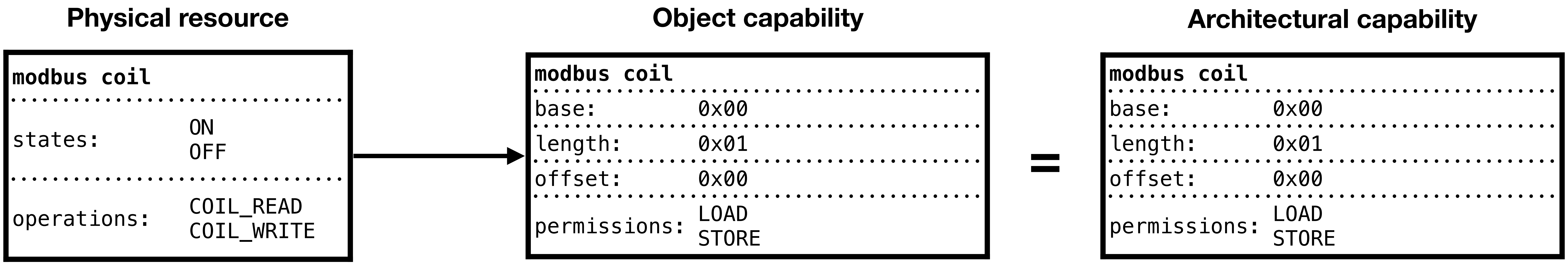
This is the layer of on-device interaction

- 'Owner' process distributes tokens to potential 'users'
- 'User' processes return token with request to 'owner'
- 'Owner' verifies the object and maps it to constituent architectural capabilities

CHERI capabilities are used abstractly

- Memory addresses used to encode information, but not store data
- E.g., `base:0x00` and `length:0x0a` encode speed settings of a motor between 0 and 10

CHERI-based object capabilities



Physical resource

modbus coil	
states:	ON OFF
operations:	COIL_READ COIL_WRITE

Object capability

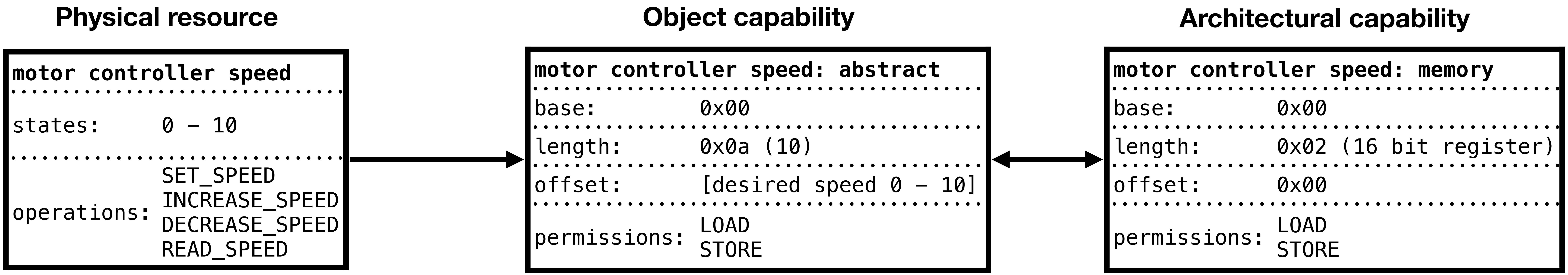
modbus coil	
base:	0x00
length:	0x01
offset:	0x00
permissions:	LOAD STORE

=

Architectural capability

modbus coil	
base:	0x00
length:	0x01
offset:	0x00
permissions:	LOAD STORE

A coil is an object that can be energised or deenergised at a given voltage to control solenoids, energise motors, trip breakers, etc.



Physical resource

motor controller speed	
states:	0 – 10
operations:	SET_SPEED INCREASE_SPEED DECREASE_SPEED READ_SPEED

Object capability

motor controller speed: abstract	
base:	0x00
length:	0x0a (10)
offset:	[desired speed 0 – 10]
permissions:	LOAD STORE

Architectural capability

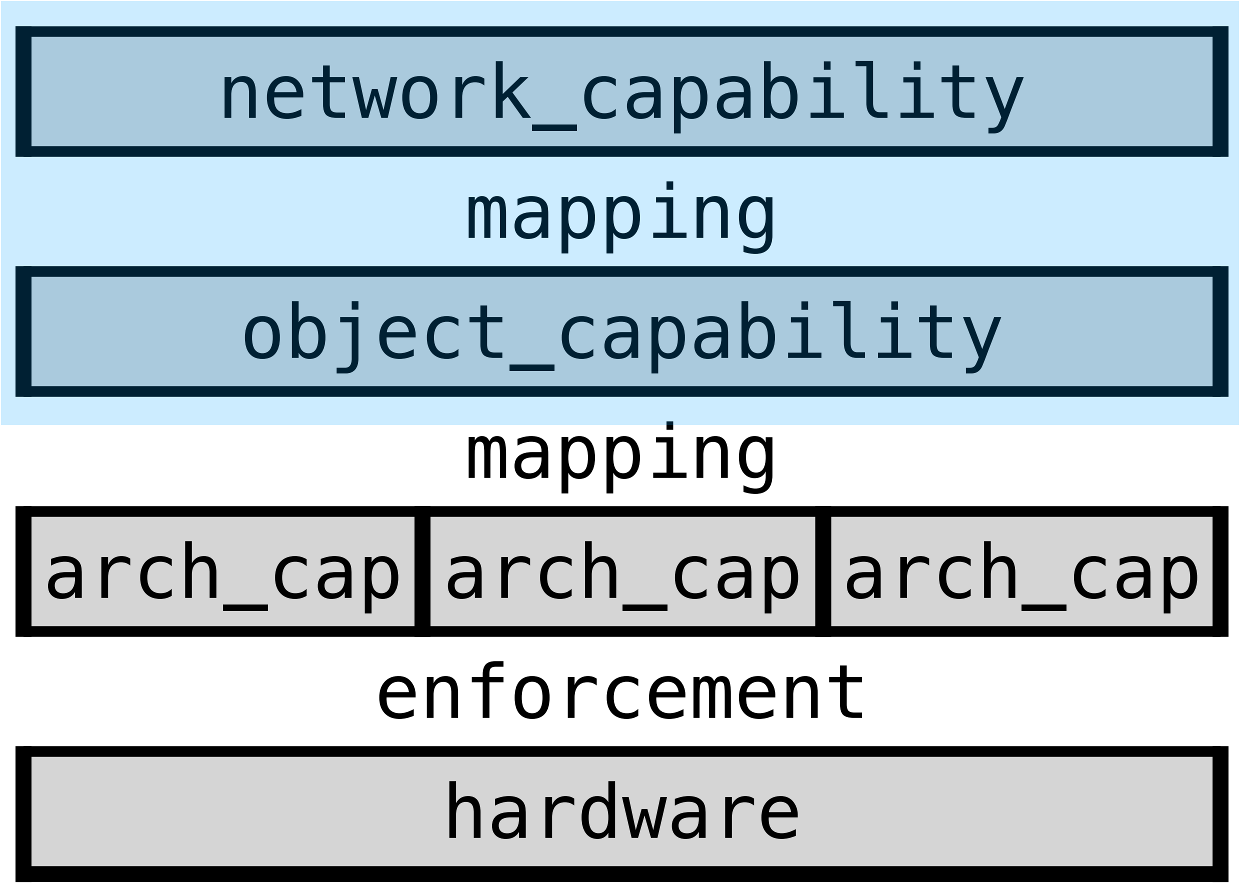
motor controller speed: memory	
base:	0x00
length:	0x02 (16 bit register)
offset:	0x00
permissions:	LOAD STORE

Motor controllers convert an intuitive input (e.g., relative speed 0 to 10) to the motor’s actual control mechanism (e.g., frequency)

Macaroon-based network capabilities

Macaroons: Bearer tokens providing efficient decentralised delegation and attenuation of privilege

Origin: Distributed authorisation in the cloud



Properties

- Key holder can initiate/verify
- Any holder can attenuate
- Protected by keyed HMAC chain

Benefits for CPS

- Limited cryptographic burden
- Ease of attenuation and delegation
- Semantic similarity to CHERI

Birgisson et al. Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud, 2014

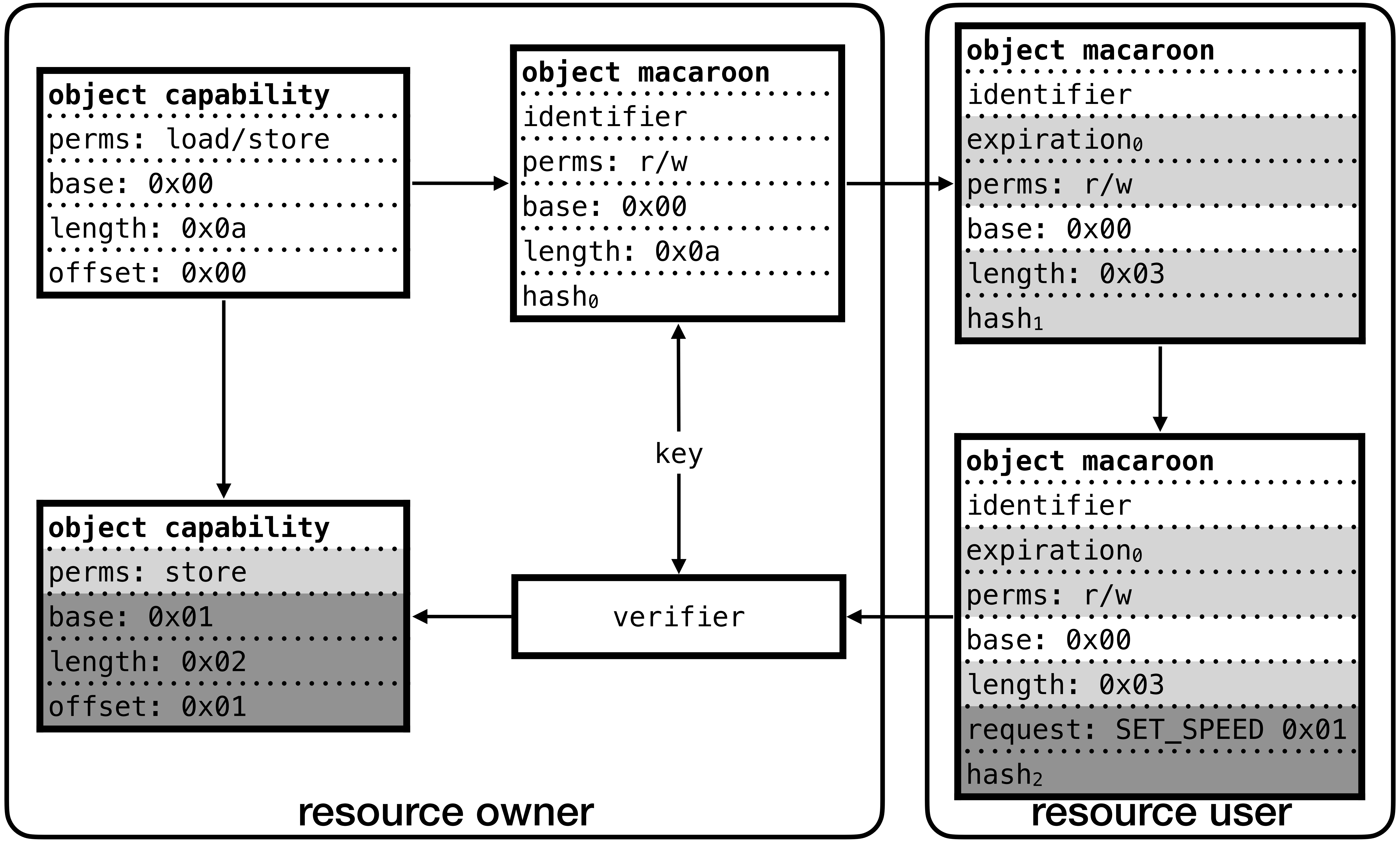
Composing CHERI and Macaroons

CHERI to Macaroons

- Map CHERI capability metadata to Macaroon caveats

Macaroons to CHERI

- Verify Macaroon and derive a new, restricted CHERI capability



CHERI Macaroons security properties

General

- Spatial memory safety
- Fine-grained access control at the host

Network

- Integrity protection for unencrypted and unauthenticated protocols

Device

- Protection against adversarial processes or tasks*
- `sudo`-like minimal privilege of the resource-owning process*

***provided CHERI compartments are implemented**

Practicalities and challenges

Hardware support

- MIPS and RISC-V FPGA cores
- Arm Morello and CHERI-ARM-M

Software support

- Memory safety is (mostly) free
- Object capabilities require software definition

Token distribution

- Requires manual installation or centralised authentication and distribution
- Examples: trust on first use, manual distribution, Kerberos

Case study: Modbus

Goal: Implement CHERI, object, and network capabilities without modifying existing code

Modbus: Ubiquitous ICS protocol commanding coils, discrete inputs, and registers

Examples:

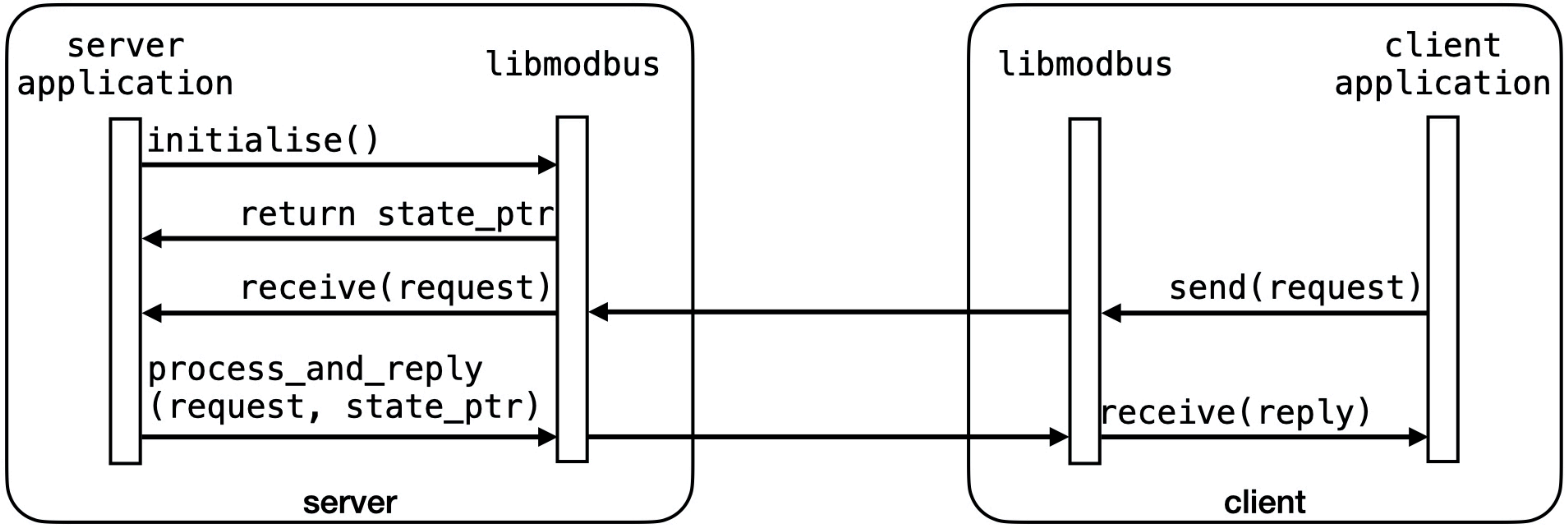
```
READ_COIL
READ_DISCRETE_INPUT
WRITE_REGISTER
```

Platform:

CheriBSD
CheriFreeRTOS

Performance:

cost << RTOS loop or network delay



Summary

Capabilities support intuitive, host-based CPS access control:

- CHERI: Efficient memory safety and basis for object capabilities in CPS
- Macaroons: Protection for legacy protocols and simple mapping to CHERI object capabilities
- CHERI Macaroons: Effective access control against strong adversaries on the hardware or the network

Ongoing CHERI compartmentalisation work:

- Trusted compartments
- Protection for intertask communication