Protecting supply chains with CHERI

David Chisnall

SCI Semiconductor

ars TECHNICA

NIGHTMARE SUPPLY CHAIN ATTACK SCENARIO —

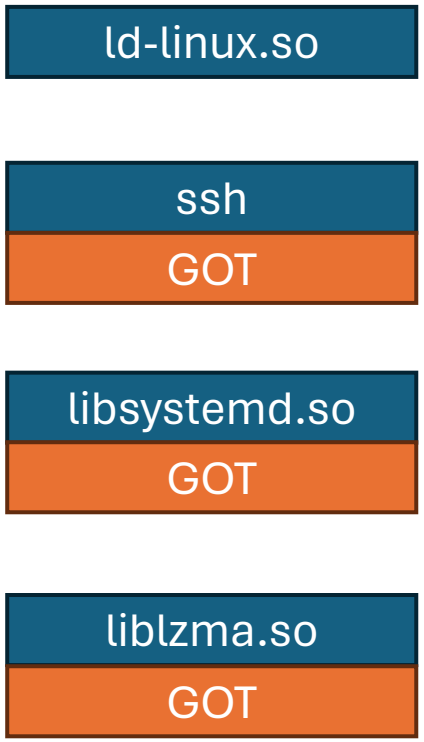
What we know about the xz Utils backdoor that almost infected the world

Malicious updates made to a ubiquitous tool were a few weeks away from going mainstream.

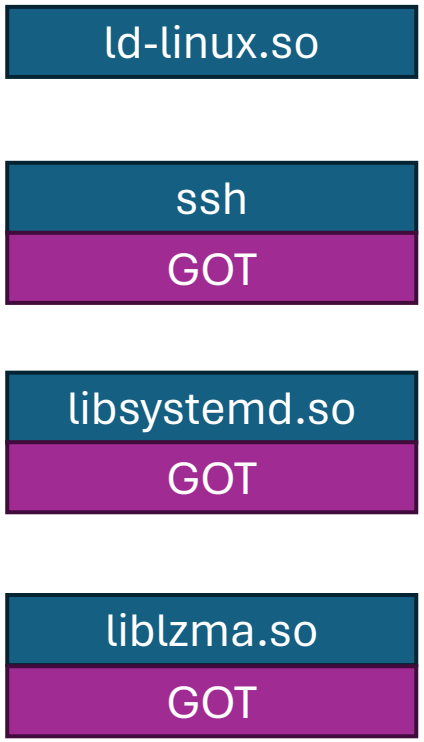
DAN GOODIN - 4/1/2024, 7:55 AM

Getty Images

What (nearly) went wrong with liblzma?

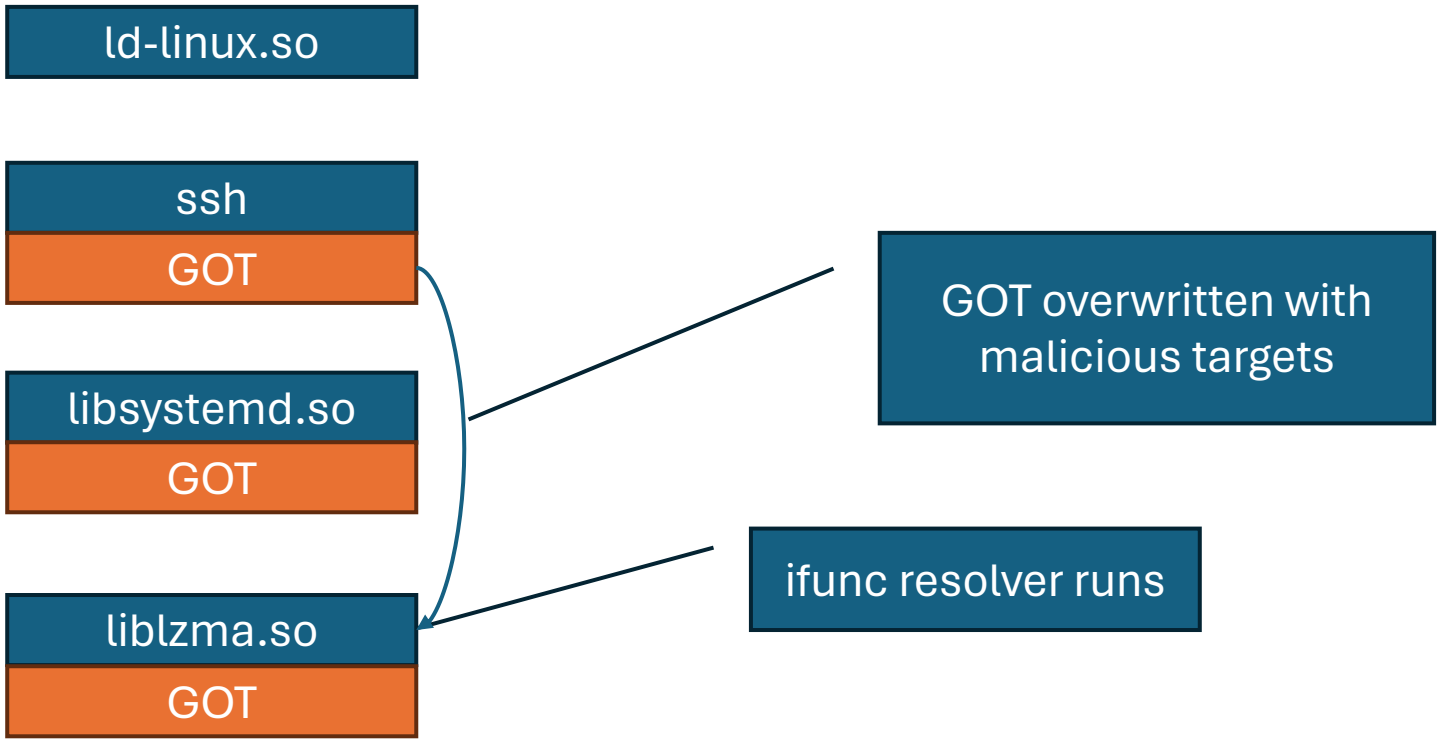


What (nearly) went wrong with liblzma?



Read-only GOTs protect cross-library control flow

What (nearly) went wrong with liblzma?



The ifunc is not the problem

Any library can change GOT permissions

Any library can tamper with any other data

What happens when a supply-chain attacker compromises your program?

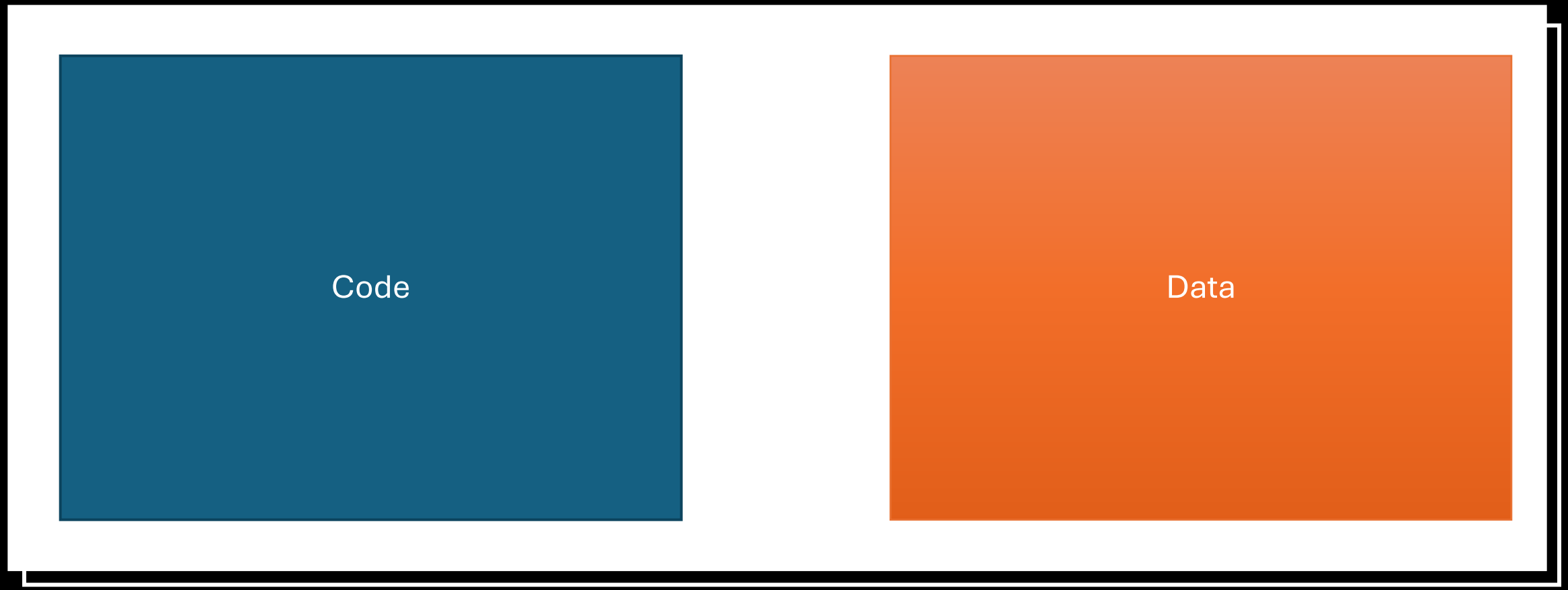
Game Over

Supply chain security requires boundaries around reused code

CHERI Compartmentalization

Mitigating Unknown Vulnerabilities

What is a compartment?



Isolation is easy, sharing is hard

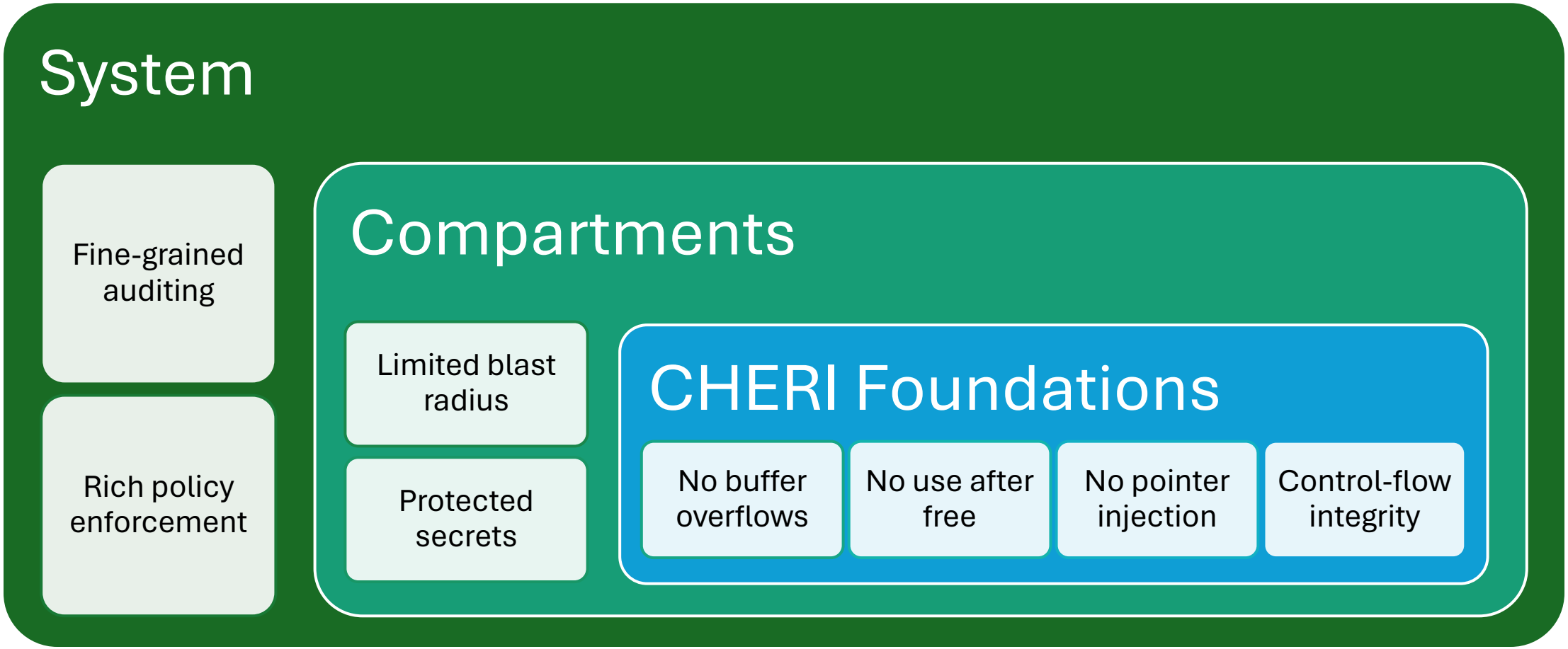
CHERI is designed to enable safe sharing!

Compartments interact only via capabilities

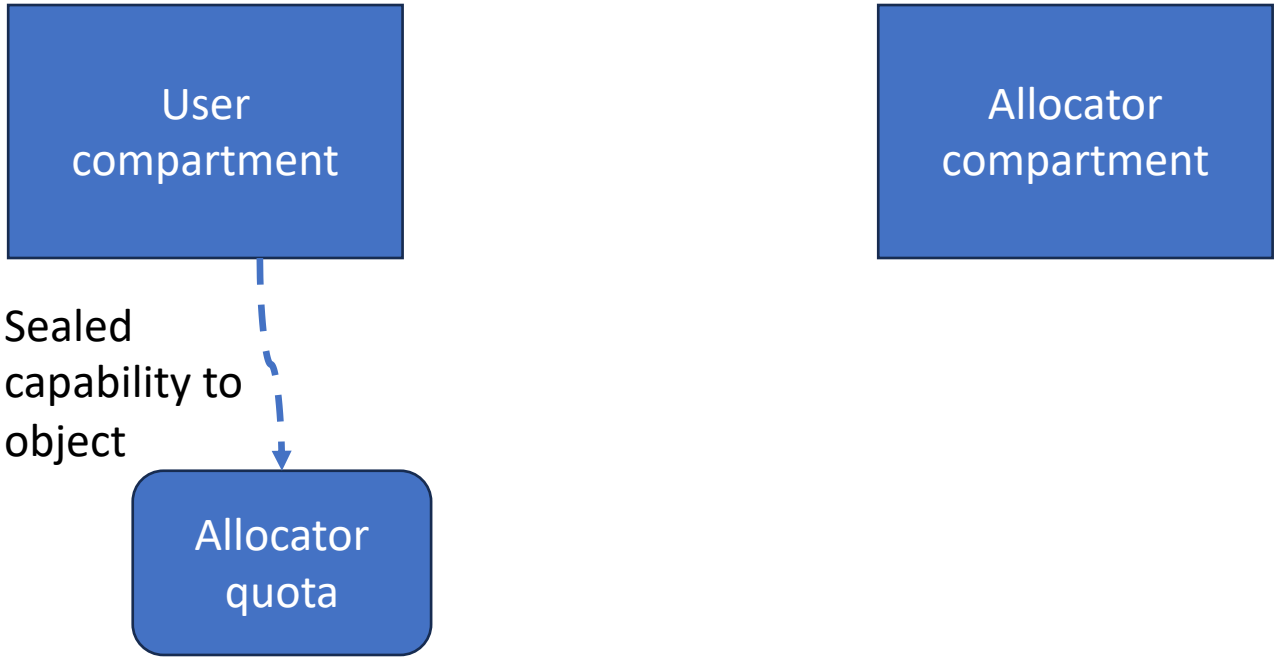
Call other compartments via call gates

Access shared resources (e.g. MMIO regions) only via memory capabilities

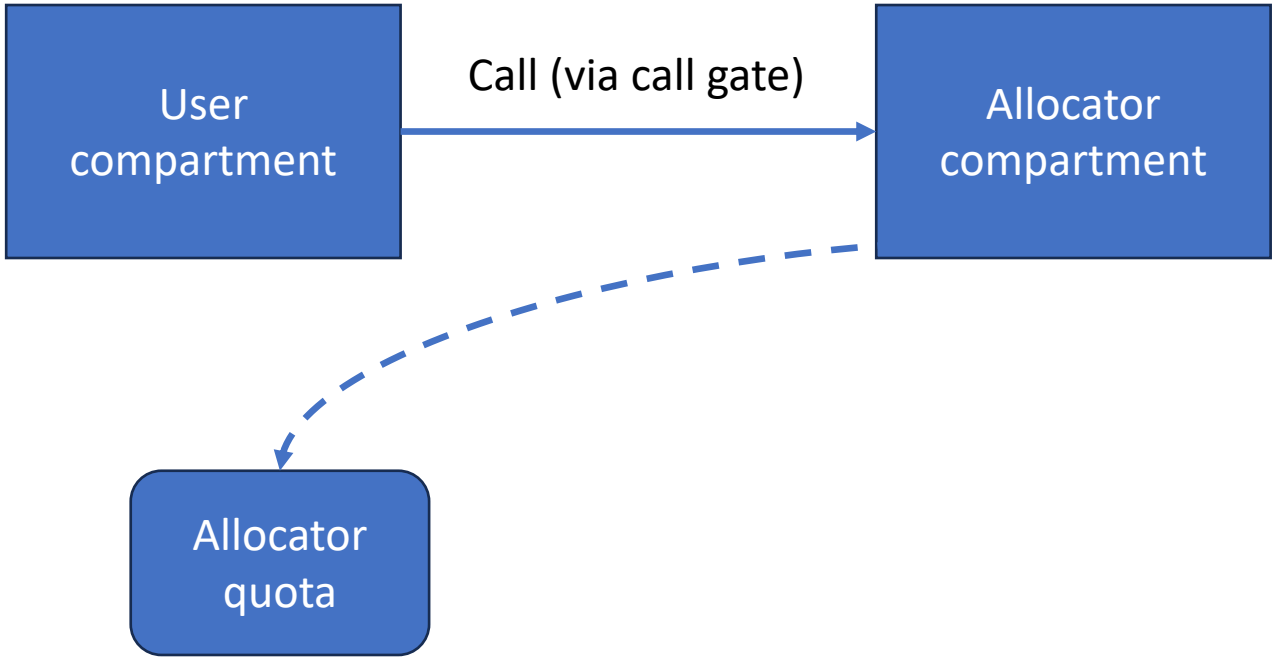
CHERIoT provides layered security



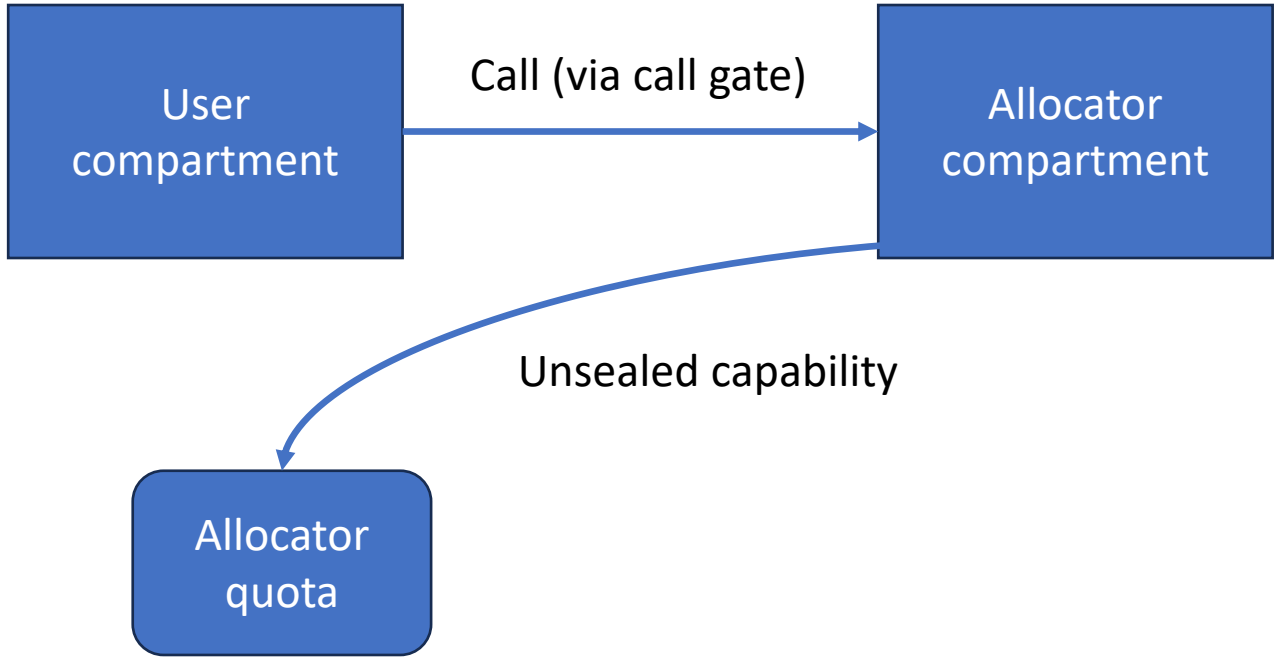
Sealing enables software capabilities



Sealing enables software capabilities



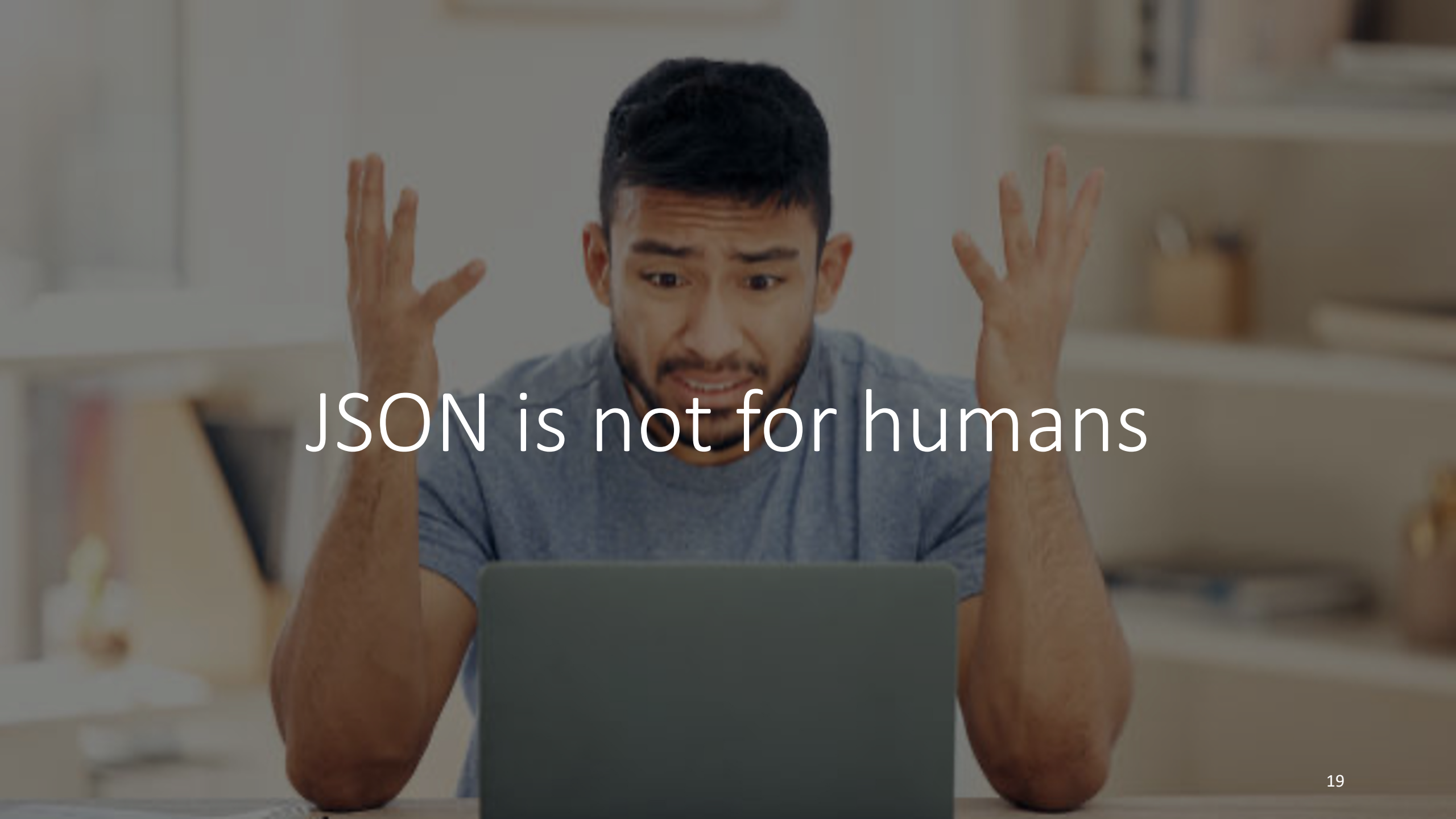
Sealing enables software capabilities



CHERIoT linker reports describe contents and interactions

- Code and data hashes
- Exported functions
- Imported functions
- MMIO regions
- Sealed objects
- Thread stack sizes
- Thread entry points


```
{
  "compartments": {
    "Firewall": {
      "code": {
        "inputs": [
          {
            "file": "build/cheriot/cheriot/release/Firewall.compartment",
            "section_name": ".text",
            "sha256": "b69e004de8cbaee30f71f5f4d929f57ed0e21401f4130ee31e108b40c93b2688",
            "size": 4850
          },
          {
            "file": "build/cheriot/cheriot/release/Firewall.compartment",
            "section_name": ".init_array",
            "sha256": "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
            "size": 0
          }
        ],
        "name": ".Firewall_code",
        "output": {
          "sha256": "eb6f4833e07c93b357411bc5328e8681c5e5ee2ba65e14f9d2894c978e195407"
        }
      },
      "exports": [
        {
          "export_symbol": "__export_Firewall__Z21ethernet_driver_startv",
          "exported": true,
          "interrupt_status": "enabled",
          "kind": "Function",
          "register_arguments": 0,
          "start_offset": 208
        },
```



JSON is not for humans

Rego policy language

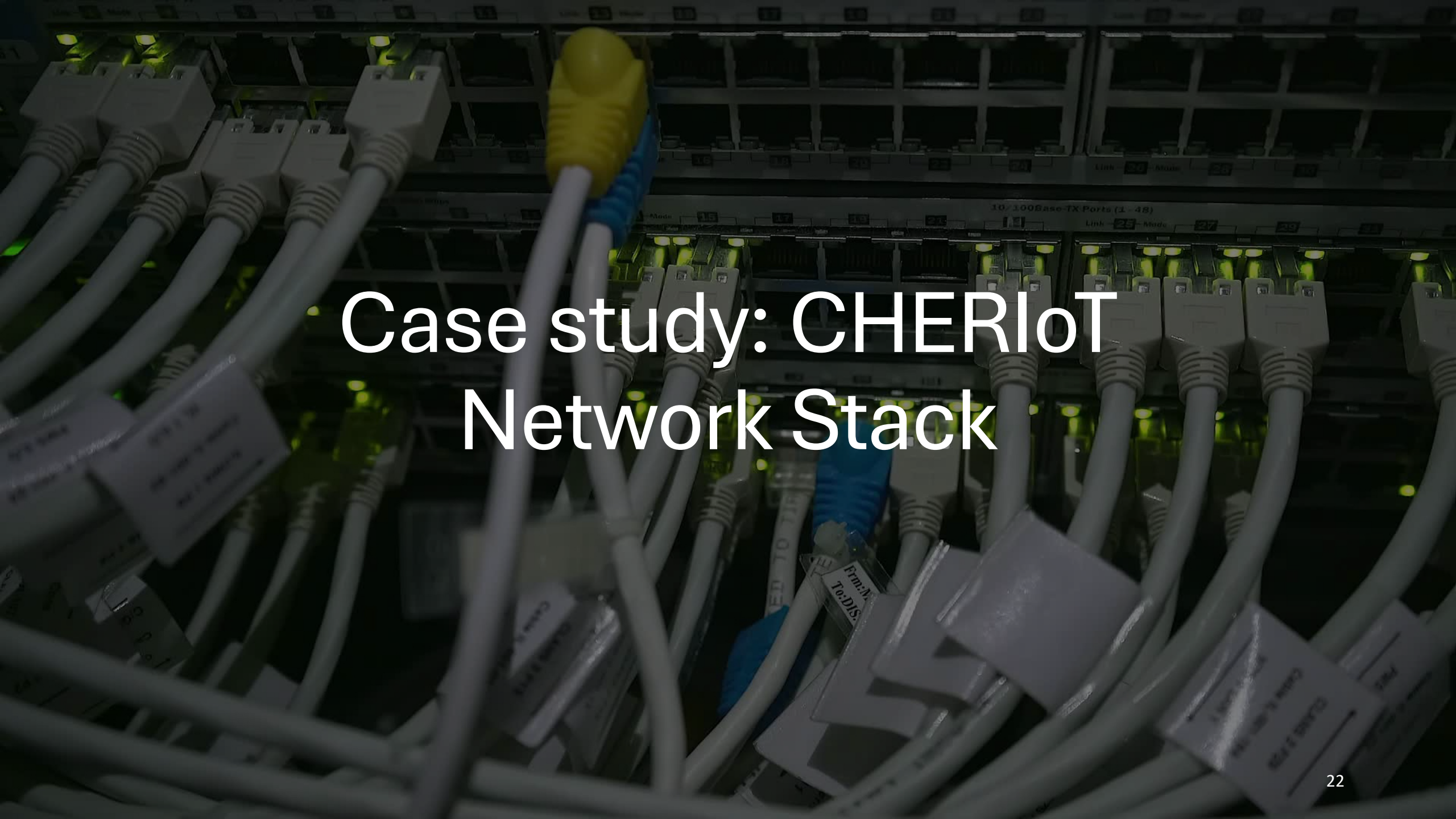
- Part of the OpenPolicyAgent project
- Mostly declarative policy language
- Consumes JSON, produces JSON
- Supports composable modules

CHERIoT-Audit consumes JSON with Rego

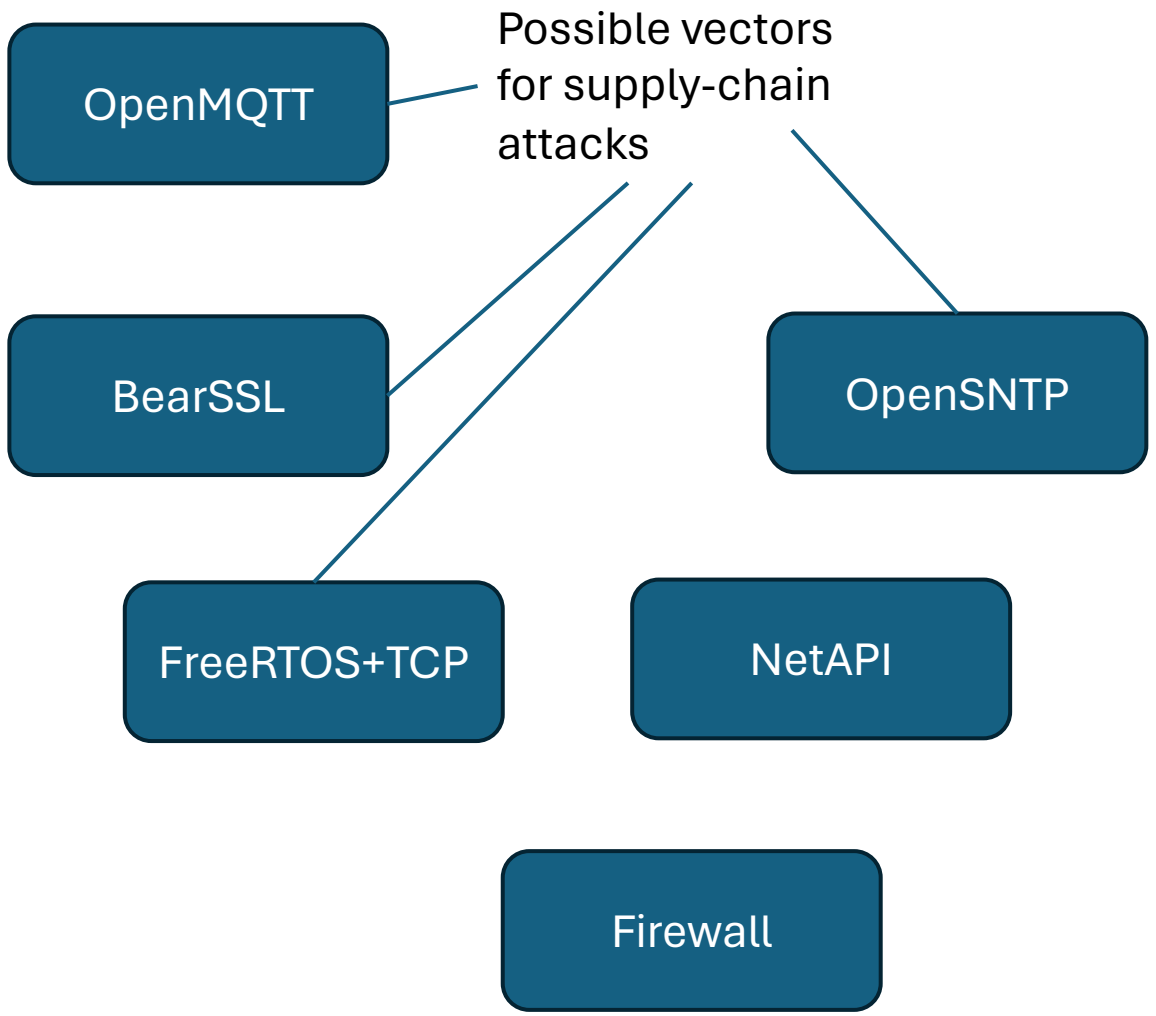
Firmware integrators write policies

`cheriot-audit` checks them

Can also inspect compartment status



Case study: CHERIoT Network Stack


Possible vectors for supply-chain attacks
OpenMQTT
BearSSL
OpenSNTP
FreeRTOS+TCP
NetAPI
Firewall

No compartment except the firewall may access the ethernet device directly

data.compartment.mmio_allow_list(ethernetDevice, {"Firewall"})

The TCP/IP compartment’s incoming frame API is exposed only to the Firewall compartment

data.compartment.compartment_call_allow_list("TCPIP", "ethernet_receive_frame.*", {"Firewall"})

data.network_stack.all_connection_capabilities

What compartments can connect where?


```
[ {
    "capability": {
      "connection_type": "UDP",
      "host": "pool.ntp.org",
      "port": 123
    },
    "owner": "SNTP"
  }, {
    "capability": {
      "connection_type": "TCP",
      "host": "cheriot.demo",
      "port": 8883
    },
    "owner": "mqtt_demo"
  } ]
```

What happens when a supply-chain attacker compromises the TCP/IP stack?

They cannot call user code

They cannot allocate more memory than their quota

They cannot control the firewall

They can tamper with packets to and from the network

They can (currently) lie about DNS responses

Summary

See https://cheriot.org for more information!

Memory safety is just the start

CHERI memory safety is a building block for compartmentalisation

Sealing is essential for rich abstractions

Compartmentalisation is a key part of supply-chain security