Secure Satellite Software-Defined Payloads with High-Assurance Post-Quantum Cryptography

K. Bhargavan, T. Gazagnaire, F. Kiefer, **V. Robles**

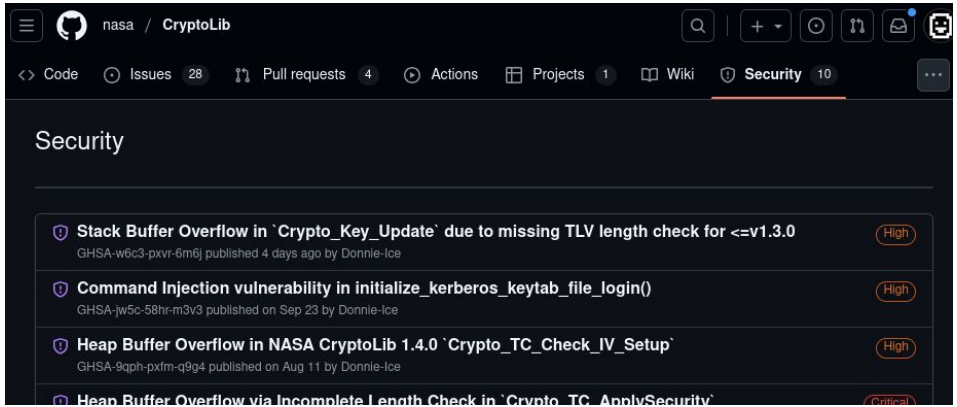
Parsimoni

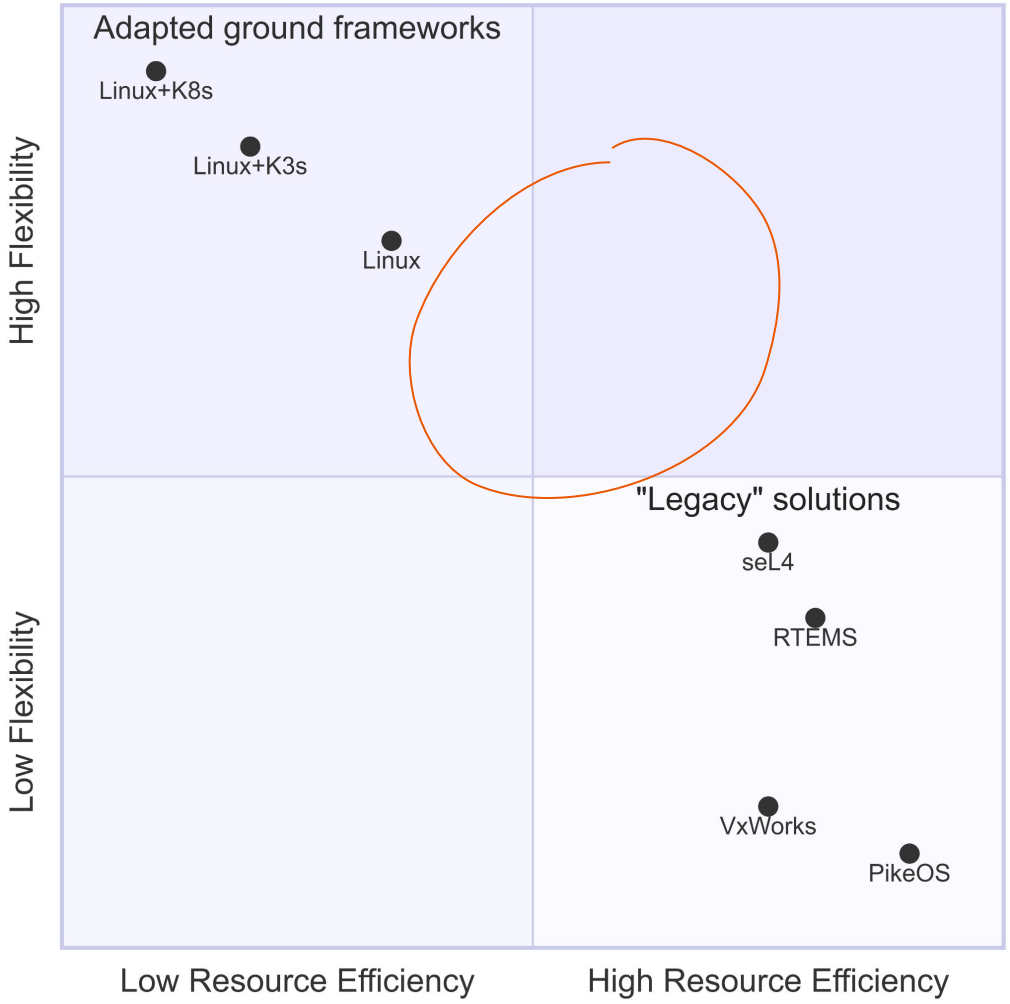
CRYSPEN

The Middleware Gap

Secure comms & cryptography

- SDLS-(EP) is fundamental, but..
  - Key distribution left to operators
  - Most implementations private
  - Doesn’t address post-quantum risks




Adapted ground frameworks
Linux+K8s
Linux+K3s
Linux
"Legacy" solutions
seL4
RTEMS
VxWorks
PikeOS
High Flexibility
Low Flexibility
Low Resource Efficiency
High Resource Efficiency

Controlled

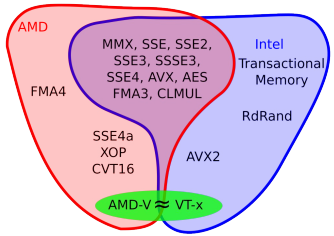
Flexible

Performant
Secure
Ad-hoc
Constrained

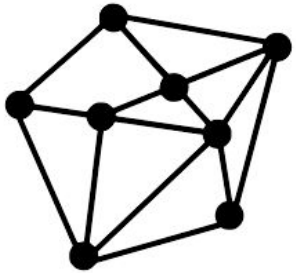
General-purpose
Reusable
Made with ground in mind
Secure?
Performance overhead

Unikernels

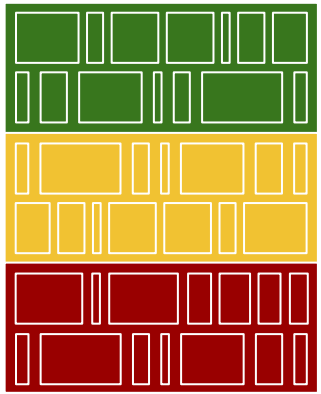
Hardware
FPGA, GPUs, etc.



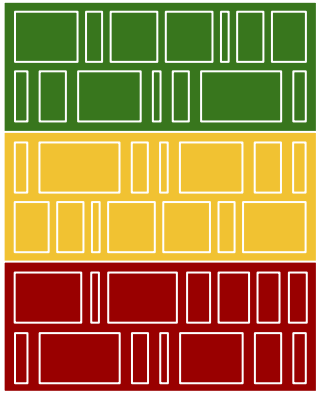
Instruction sets
RISC/CISC , AVX, etc.



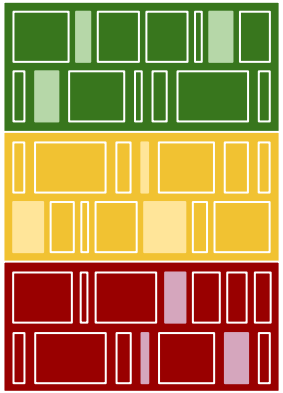
Software
Networking, graphs, etc.



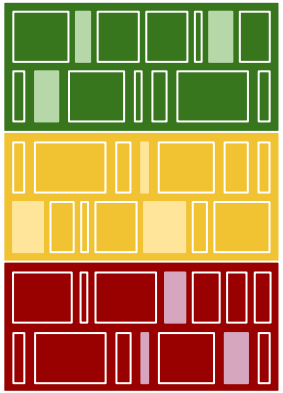
Typical OS :
general-purpose, “non-optimal”



Typical OS :
general-purpose, “non-optimal”



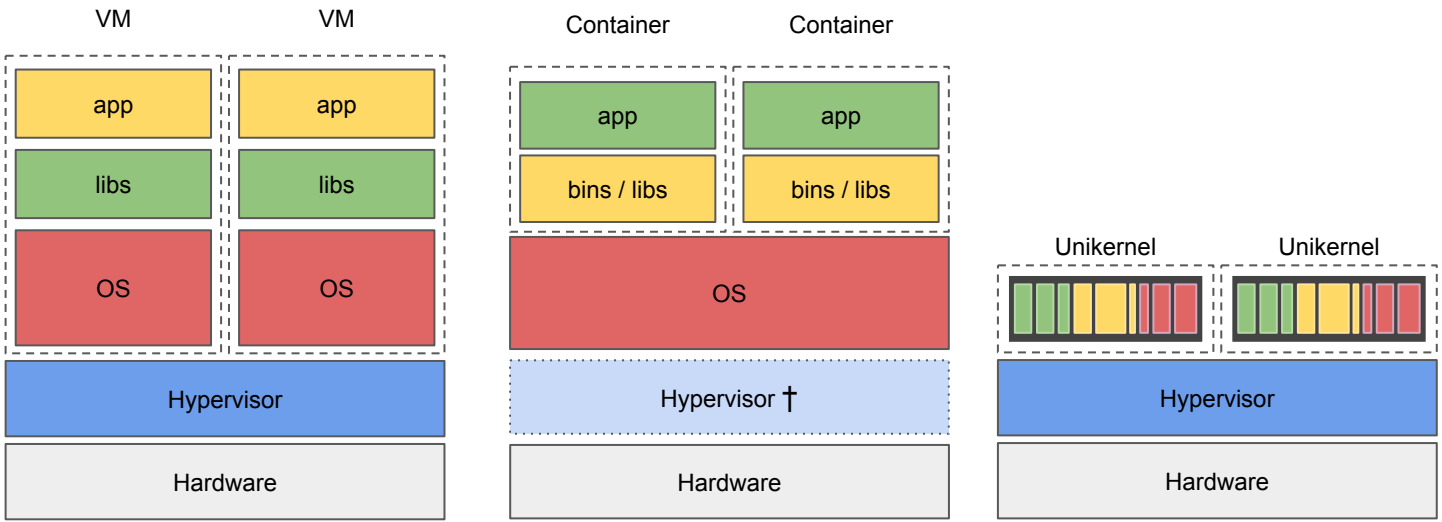
Known user-space:
dependency selection



Known user-space:
dependency selection

UNIKERNEL

Specialized micro-VM



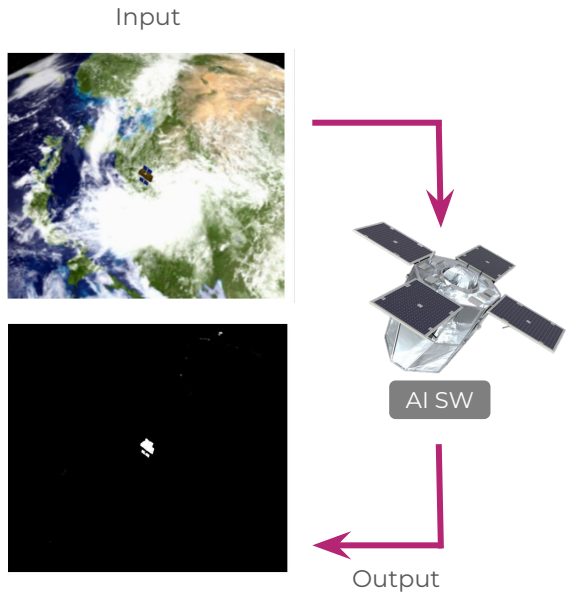
Full OS, virtualized

Containers

Unikernels

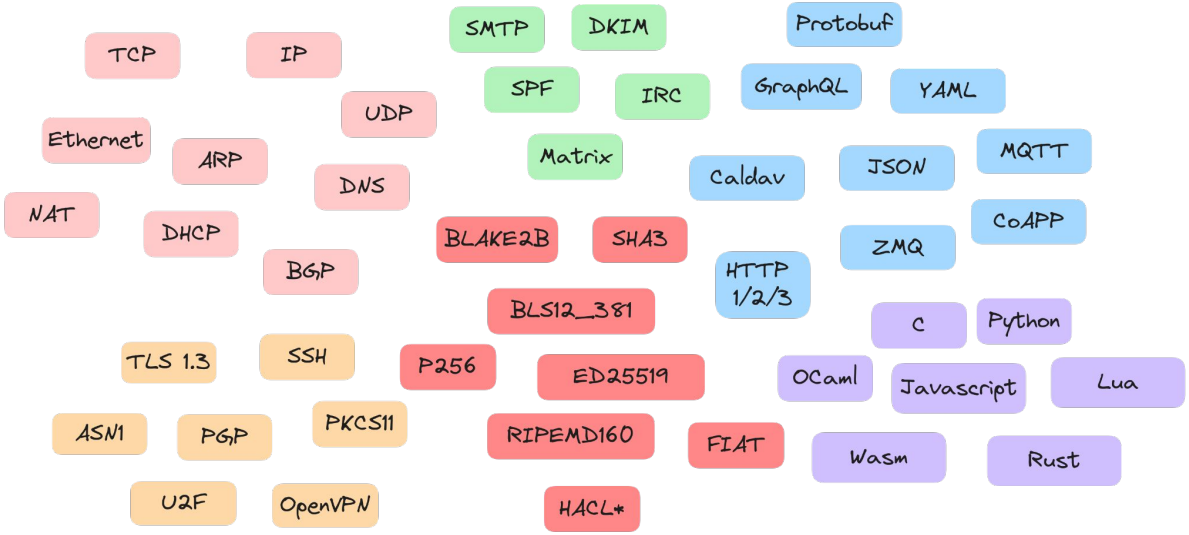
Port an image inference application

- 100% replication
- 20x smaller binary size
- 2.5x smaller memory footprint
- 20% faster runtime
- Easy to deploy and update



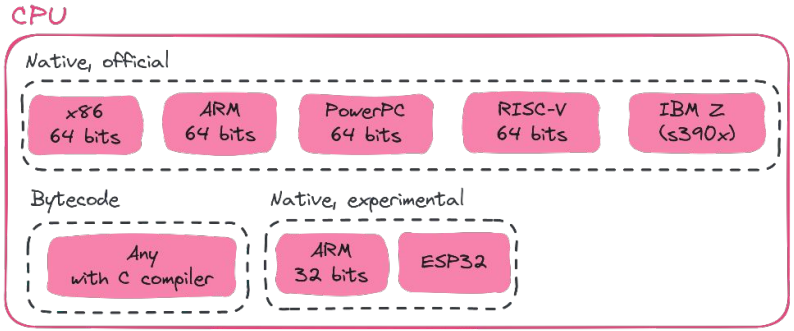
OS-as-library and security

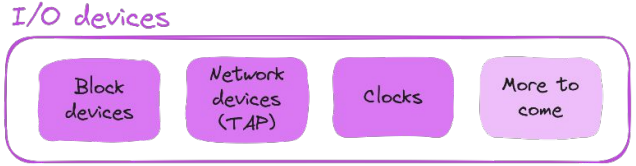
Interfaces and implementations

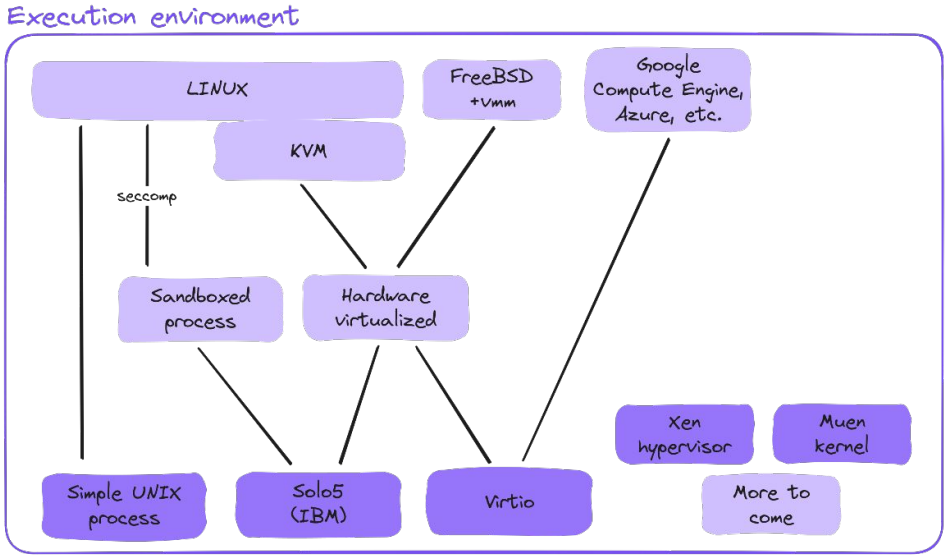


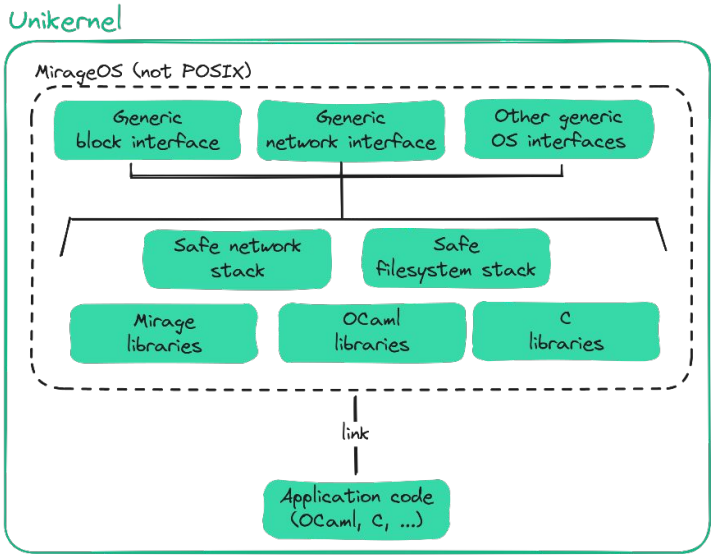
- POSIX + Linux ABI
- Performant but less portable
- C/C++

- **Clean slate interfaces**
- Portable but less performant
- OCaml + Formal Methods


CPU
Native, official
x86 64 bits
ARM 64 bits
PowerPC 64 bits
RISC-V 64 bits
IBM Z (s390x)
Bytecode
Any with C compiler
Native, experimental
ARM 32 bits
ESP32


I/O devices
Block devices
Network devices (TAP)
Clocks
More to come


Execution environment
LINUX
FreeBSD +vmm
Google Compute Engine, Azure, etc.
KVM
seccomp
Sandboxed process
Hardware virtualized
Xen hypervisor
Muen kernel
Simple UNIX process
Solo5 (IBM)
Virtio
More to come


Unikernel
MirageOS (not POSIX)
Generic block interface
Generic network interface
Other generic OS interfaces
Safe network stack
Safe filesystem stack
Mirage libraries
OCaml libraries
C libraries
link
Application code (OCaml, C, ...)

Use cases

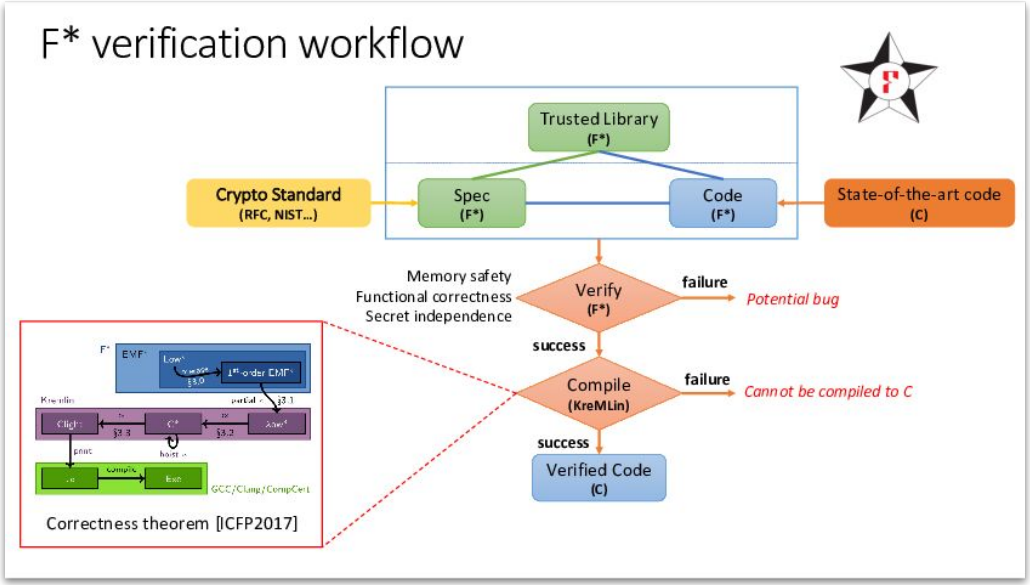
- Networking and cryptography = simple module in unikernel
  - Updatable
  - Customizable
  - **Replaceable** along well-defined interfaces

- Within host
  - Need for secure channel establishment
  - Need for secure updates of unikernels

Open-source, formally verified cryptography

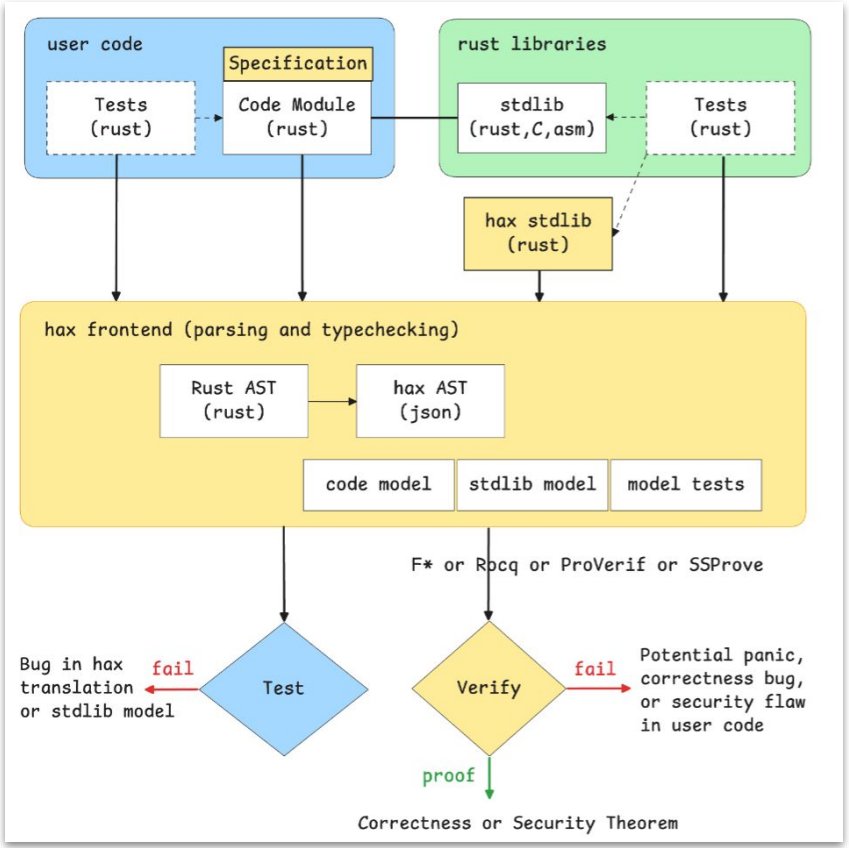
HACL/HACL*

- Formally verified cryptographic primitives
  - AEAD, ECDH, Signatures, Hashes, KDF, Ciphers, MACS
- Written, specified, verified in F*
- Compiled to C



Libcrux

- High-level, unified cryptographic Rust library
- Supports both classic and post-quantum crypto (ML-KEM, ML-DSA)
- Formally verified
  - Brings together verified artifacts (inc. HACL*)
  - Itself verified with the HAX toolchain: **runtime safe, no panic, secret independent**



Libcrux usage

Either:

- Use Rust as-is
  - Signal
- Compile down to C
  - Firefox, OpenSSH, Linux, WireGuard, ARM mbed, Python, etc.
- Cryptographic provider for other libraries

BERTIE

- **TLS 1.3 implementation**
- Formally verified
  - Runtime safe, no panic
  - Correctness of serialization, parsing
  - Safe from classes of symbolic protocol attacks
- Post-quantum safe

HAX

Opportunities for integration with unikernels

Use cases

- Key Establishment for SDLS-EP
  - Use libcrux/BERTIE to bootstrap SDLS comms
- Authenticated Channels for Quantum Key Distribution
  - ML-DSA/ML-KEM-based channel to distribute keys
- **Signed software updates**
  - ML-DSA to ensure unikernel authenticity
- **Secure channels between payloads and users**
  - Make BERTIE available to unikernels

Signed Software Updates

- Unikernels ensure isolation of software payloads
- Management of unikernels is a security-critical component

- Implemented as Proof-of-Concept in SpaceOS distribution component
  - OCaml bindings available

Secure channels between payloads and users

- Leverage MirageOS' high-level interfaces and type system
  - Libcrux as alternative crypto provider
  - BERTIE as alternative TLS implementation
- Whole crypto stack remains patchable with a software update
- Allow payload developers to choose and customize their stack

Conclusion

Conclusion

- Memory safety, and formal methods[1]

- Security opportunity of unikernels, and their clean slate interfaces

- Post-quantum cryptography, and open implementations
  - Next: actually tackle SDLS

[1]Read “A Manifesto for Applicable Formal Methods”, M. Gleirscher, J. van de Pol, J. Woodcock