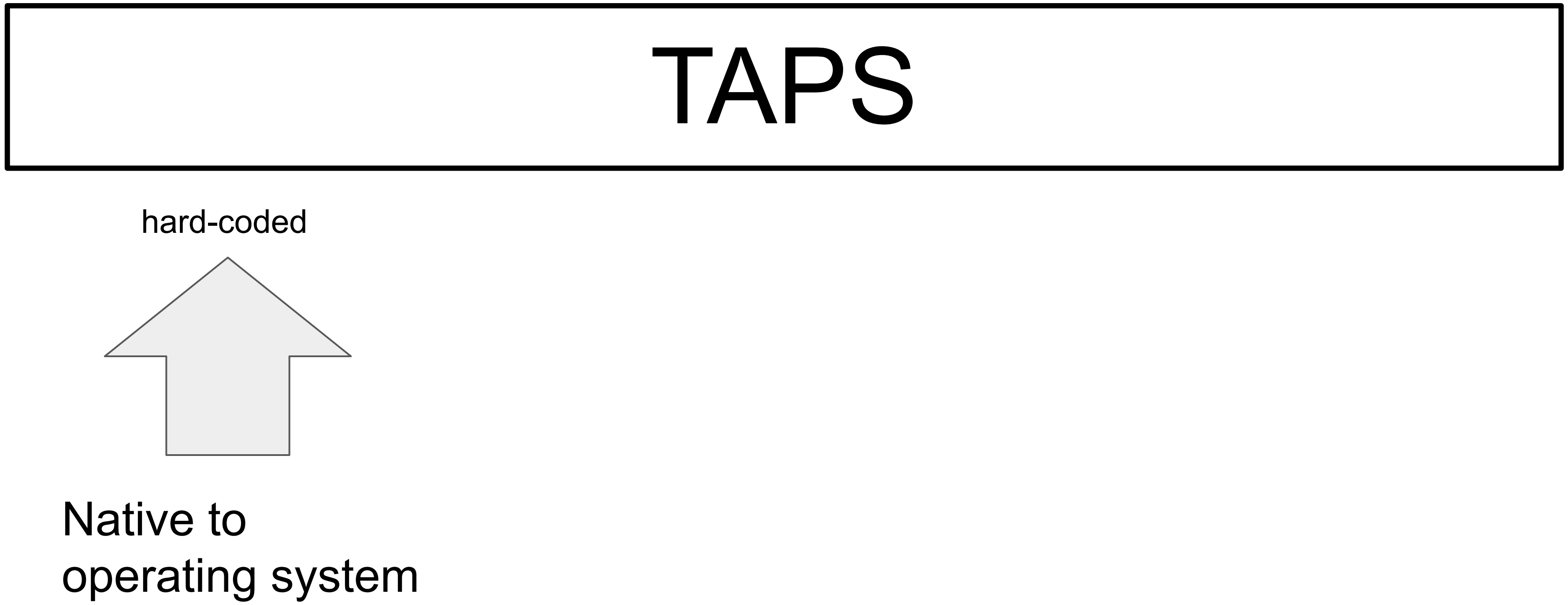
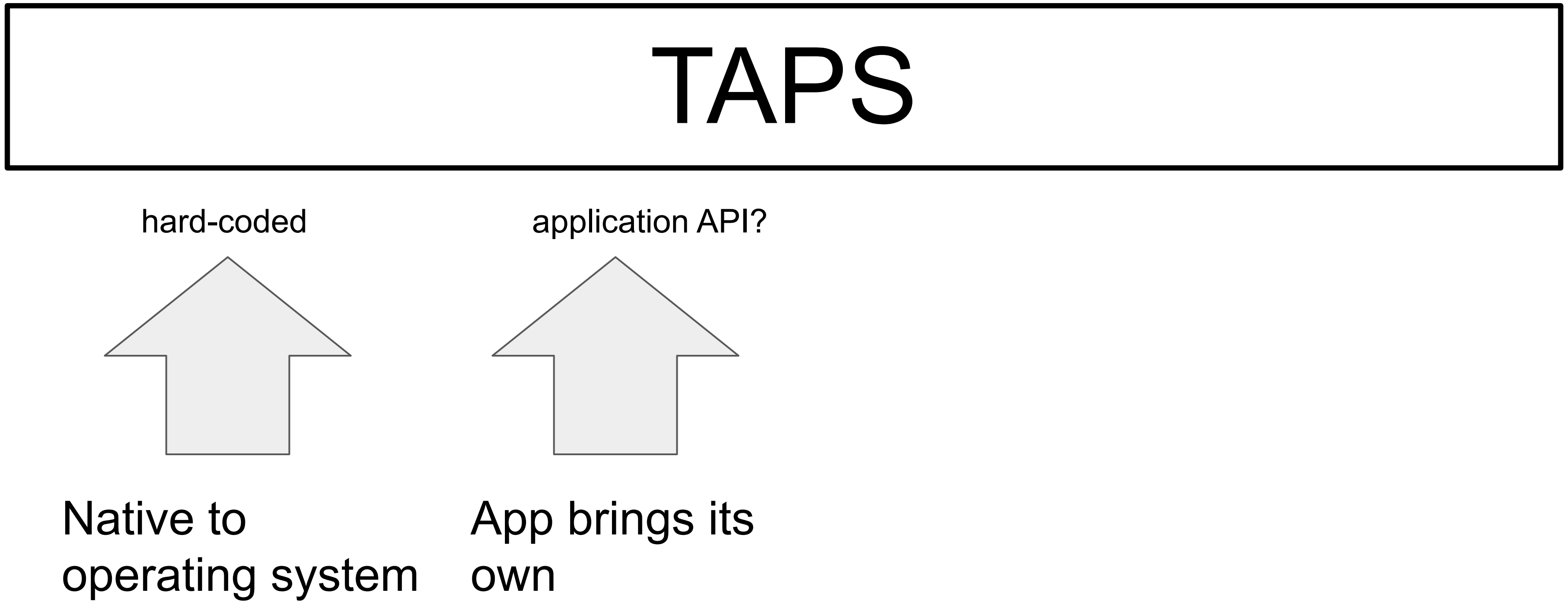
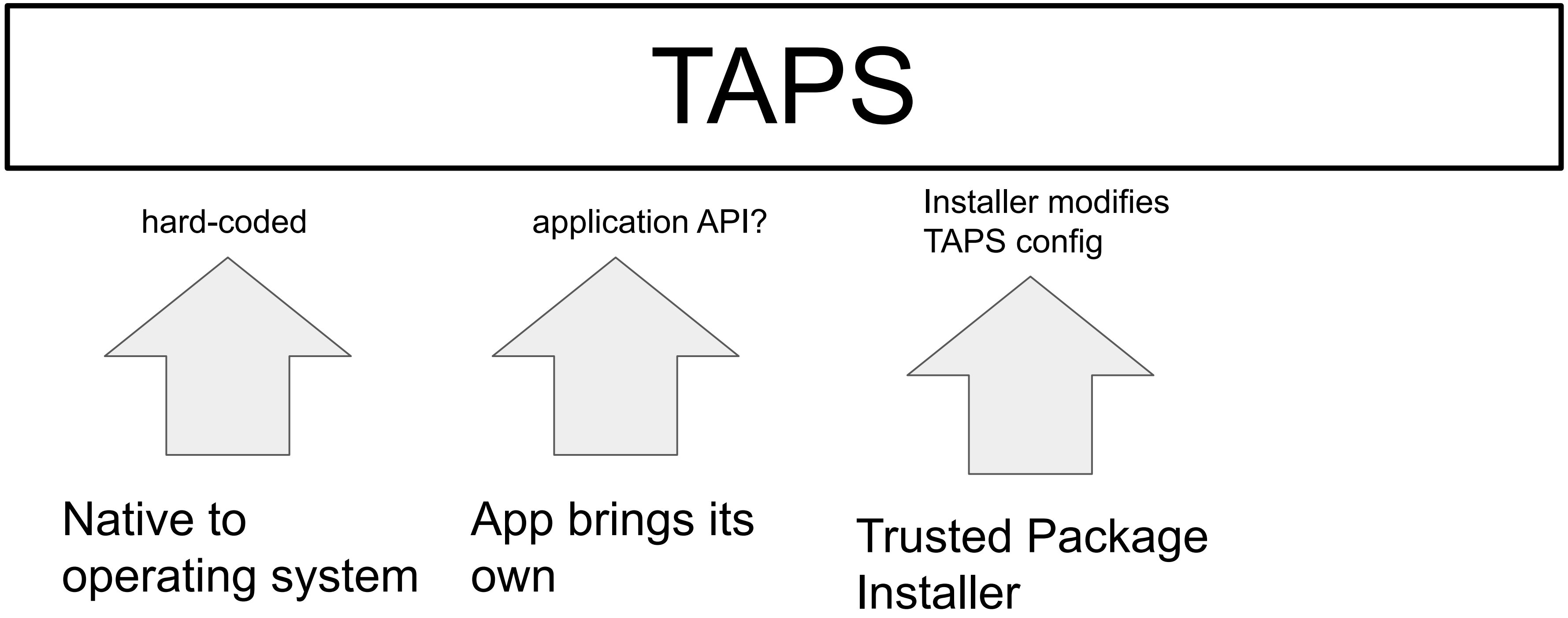
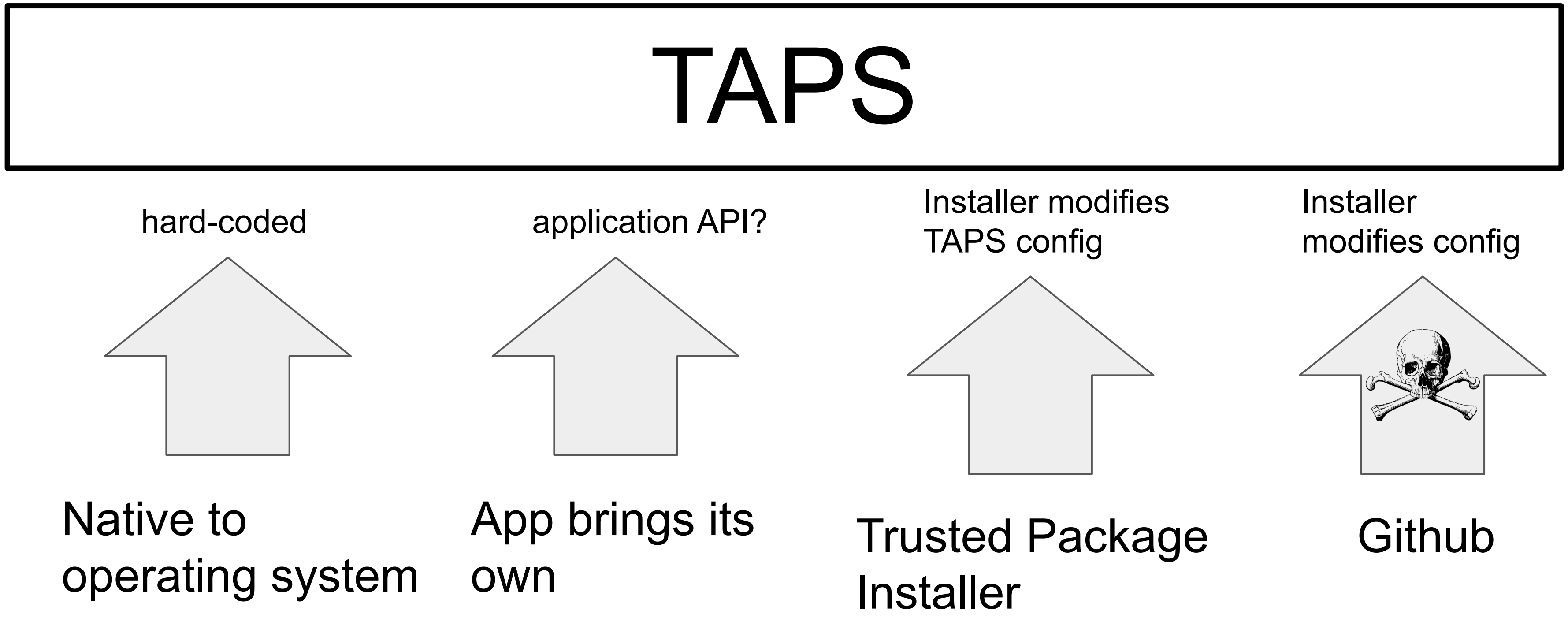
TAPS Protocol Discovery

Do we want to solve this problem?
draft-duke-taps-transport-discovery-00


TAPS
hard-coded
Native to operating system


TAPS
hard-coded
application API?
Native to operating system
App brings its own

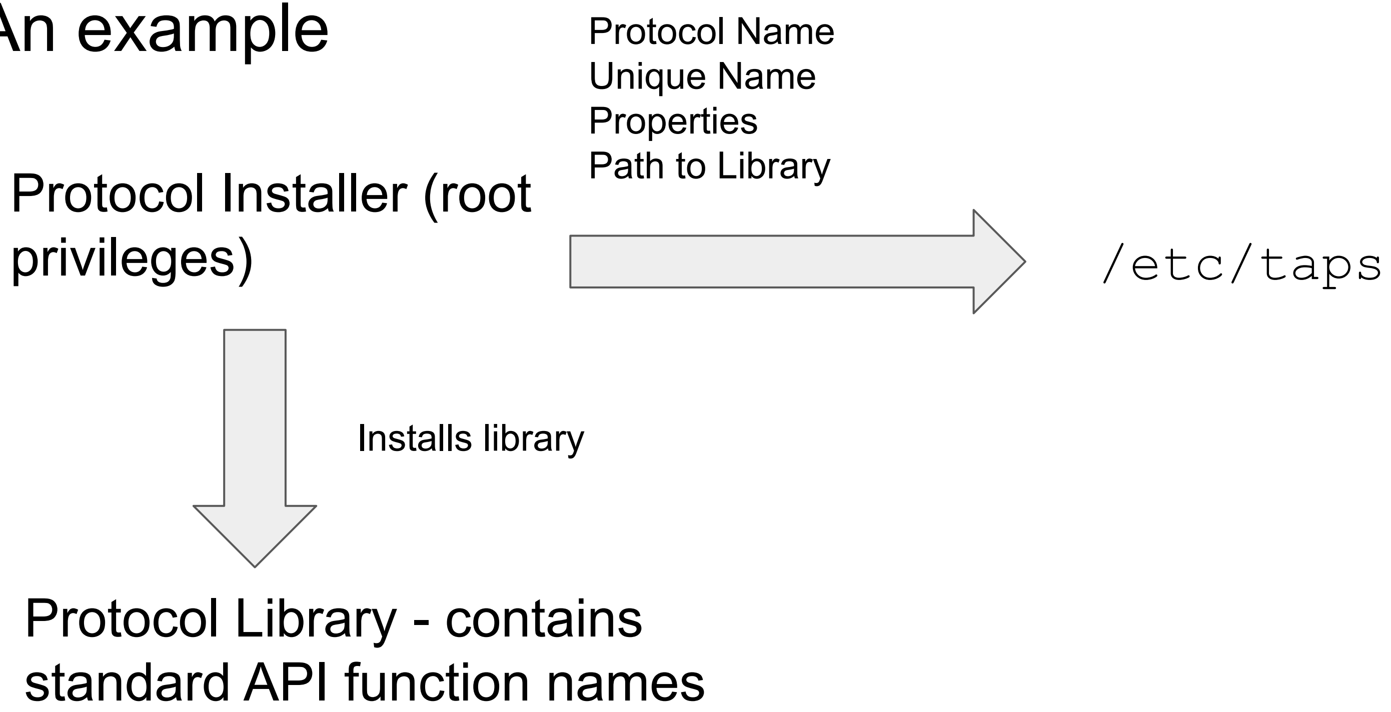

TAPS
hard-coded
application API?
Installer modifies TAPS config
Native to operating system
App brings its own
Trusted Package Installer


TAPS
hard-coded
Native to operating system
application API?
App brings its own
Installer modifies TAPS config
Trusted Package Installer
Installer modifies config
Github

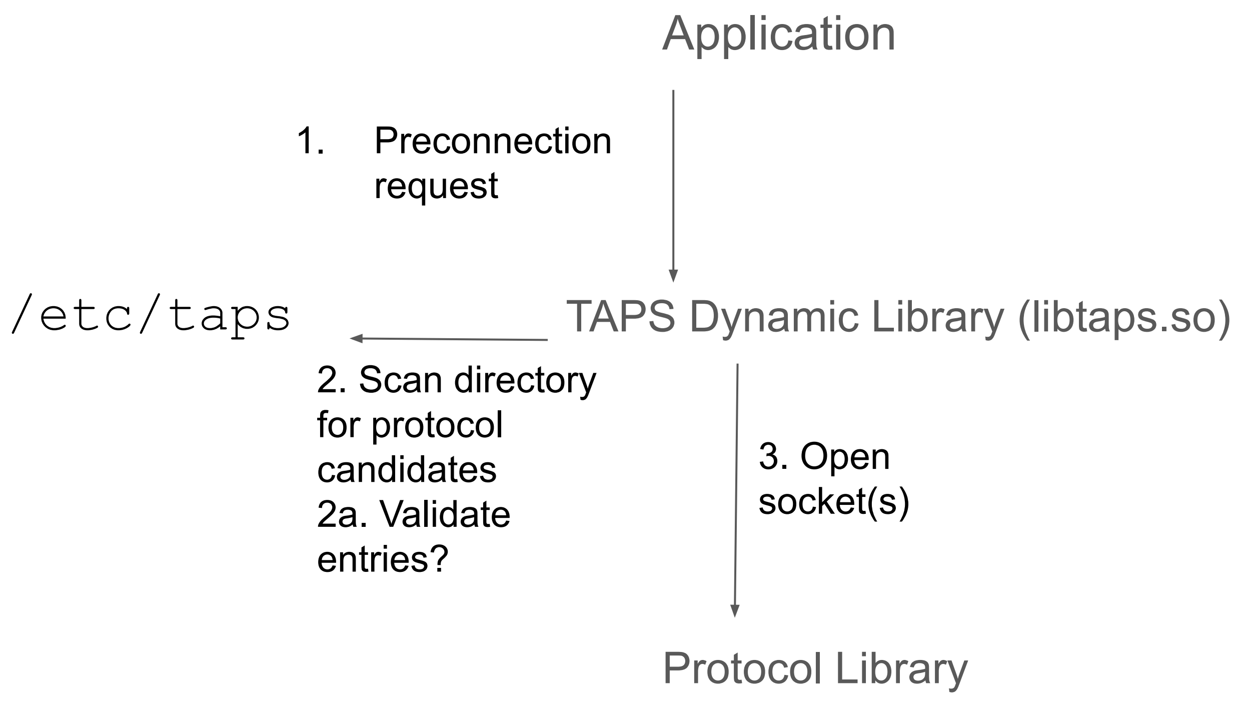
Sad outcomes for TAPS

- recompiles every time there’s a new protocol
- can’t use user space stuff
- power users can’t use bleeding edge code
  - corollary: can’t use common libraries (e.g. openssl) until that project supports TAPS
- security vulnerabilities for ordinary users
- apps have to bring their own protocols

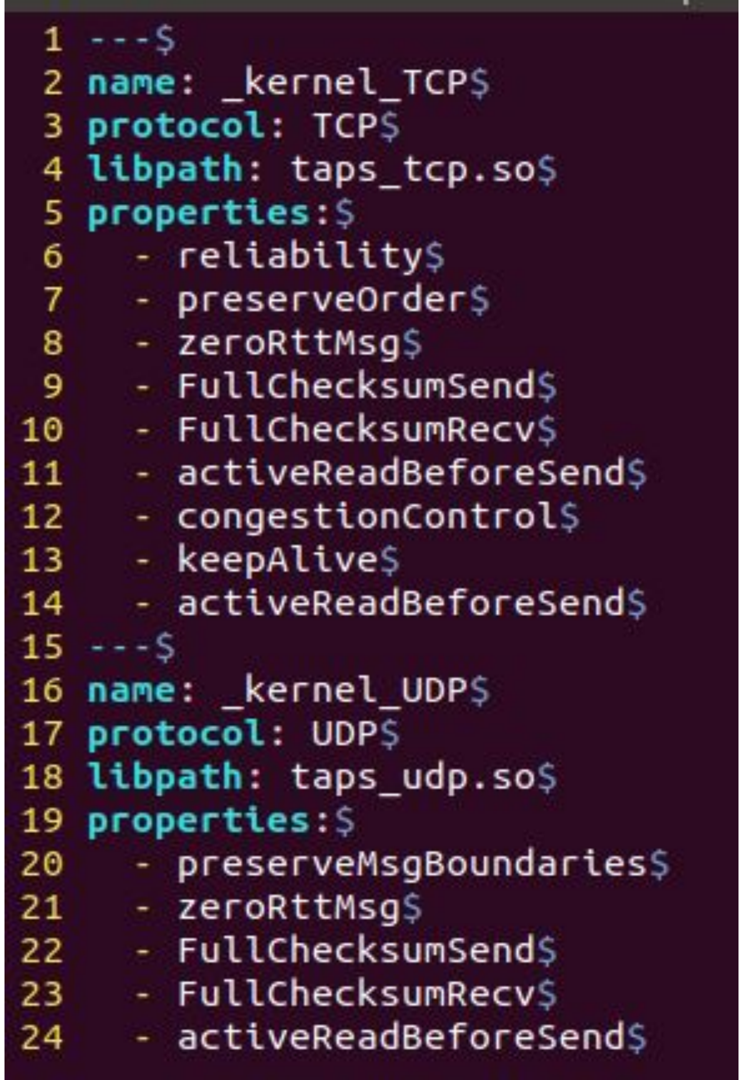
An example


Protocol Name
Unique Name
Properties
Path to Library
Protocol Installer (root privileges)
/etc/taps
Installs library
Protocol Library - contains standard API function names

An example (2)


Application
1. Preconnection request
/etc/taps
TAPS Dynamic Library (libtaps.so)
2. Scan directory for protocol candidates
2a. Validate entries?
3. Open socket(s)
Protocol Library

kernel.yaml



```
---$
name: _kernel_TCP$
protocol: TCP$
libpath: taps_tcp.so$
properties:$
  - reliability$
  - preserveOrder$
  - zeroRttMsg$
  - FullChecksumSend$
  - FullChecksumRecv$
  - activeReadBeforeSend$
  - congestionControl$
  - keepAlive$
  - activeReadBeforeSend$
---$
name: _kernel_UDP$
protocol: UDP$
libpath: taps_udp.so$
properties:$
  - preserveMsgBoundaries$
  - zeroRttMsg$
  - FullChecksumSend$
  - FullChecksumRecv$
  - activeReadBeforeSend$
```

What now?

- I’m writing some code here -- open sourcing to come
- Adopt? Does this need rechartering?