TAPS Protocol Discovery

Do we want to solve this problem?

draft-duke-taps-transport-discovery-00
TAPS

Native to operating system

hard-coded
TAPS

Native to operating system

hard-coded

App brings its own

application API?
TAPS

1. **Native to operating system**
2. **App brings its own**
3. **Installer modifies TAPS config**

- hard-coded
- application API?
- Trusted Package Installer
Native to operating system

App brings its own

Installer modifies TAPS config

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 Installer (root privileges)

Installs library

Protocol Library - contains standard API function names

Protocol Name
Unique Name
Properties
Path to Library

/etc/taps
An example (2)

1. Preconnection request
2. Scan directory for protocol candidates
   2a. Validate entries?
3. Open socket(s)

Application

/etc/taps

TAPS Dynamic Library (libtaps.so)

Protocol Library
- 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?