Substrate Protocol for User Datagrams (SPUD) Prototype
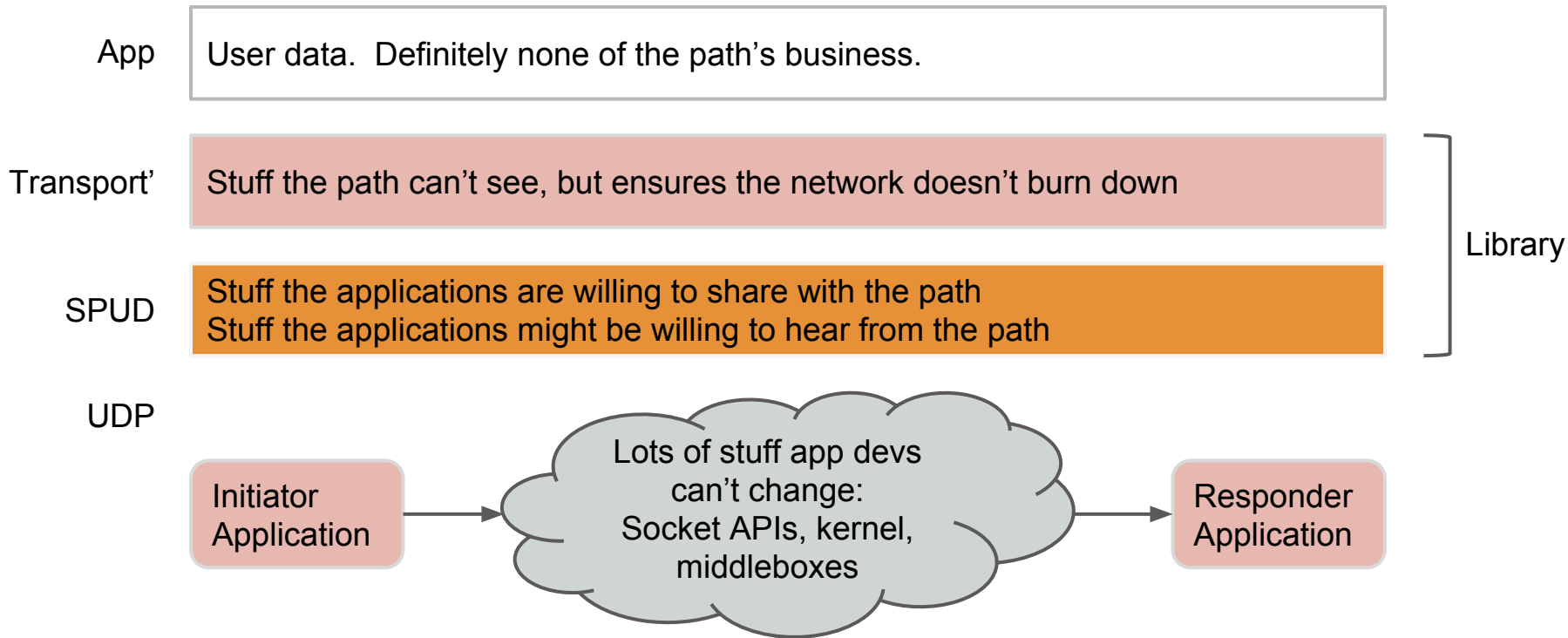
draft-hildebrand-spud-prototype

Joe Hildebrand
2015-03-25

Architecture

App	User data. Definitely none of the path's business.
Transport'	Stuff the path can't see, but ensures the network doesn't burn down
SPUD	Stuff the applications are willing to share with the path Stuff the applications might be willing to hear from the path
UDP	

Library

Initiator Application → Lots of stuff app devs can't change: Socket APIs, kernel, middleboxes → Responder Application

Packet Layout

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	magic = 0xd80000d8																															
4	32	tube id																															
8	64																																
12	96	cmd		adec	pdec	resv				CBOR map ...																							

Magic: Constant
Command: Data, Open, Close, Ack
Application Declaration: CBOR contains information for the path bound to the tube ID
Path Declaration: CBOR contains information from the path bound to the tube ID
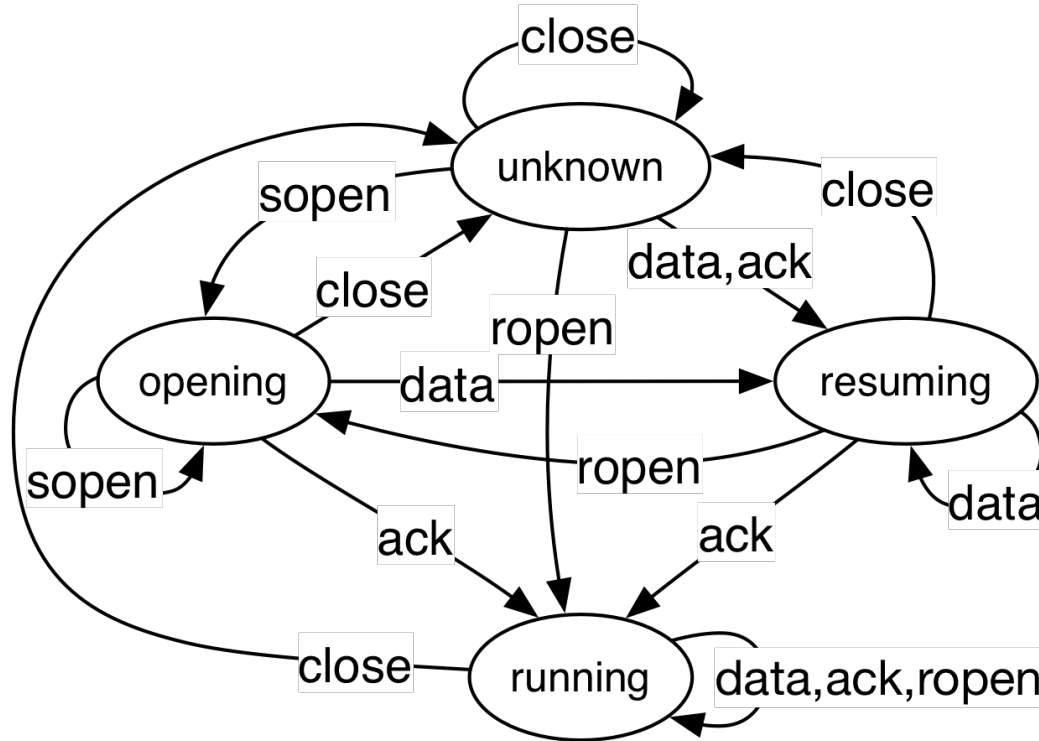**CBOR** ([RFC 7049](#)): { 0: [application data], "something else": "for the path!" }

Magic Number

Intent: minimize existing traffic as false positive match

Invalid UTF-* string and UDP protocols:

- DNS
- NTP
- DTLS
- DHCP
- SNMP
- TFTP
- Bittorrent uTP
- netbios-ns

States (as currently known)

Transport' layer (re)sends these as needed

Path has simplified view, creating state on open, beginning to destroy state on close

Declarations

Only things that must be path-visible!

Most currently in draft to be removed

Intent: minimum set needed for interop,
  all others in extension docs

Open source beginnings

IPtube (https://github.com/iptube)

SPUDlib: plain C, create tubes

SPUDshark: Wireshark dissector

TubeNode: Linux user-space netfilter

BSD license. Pull requests welcome.