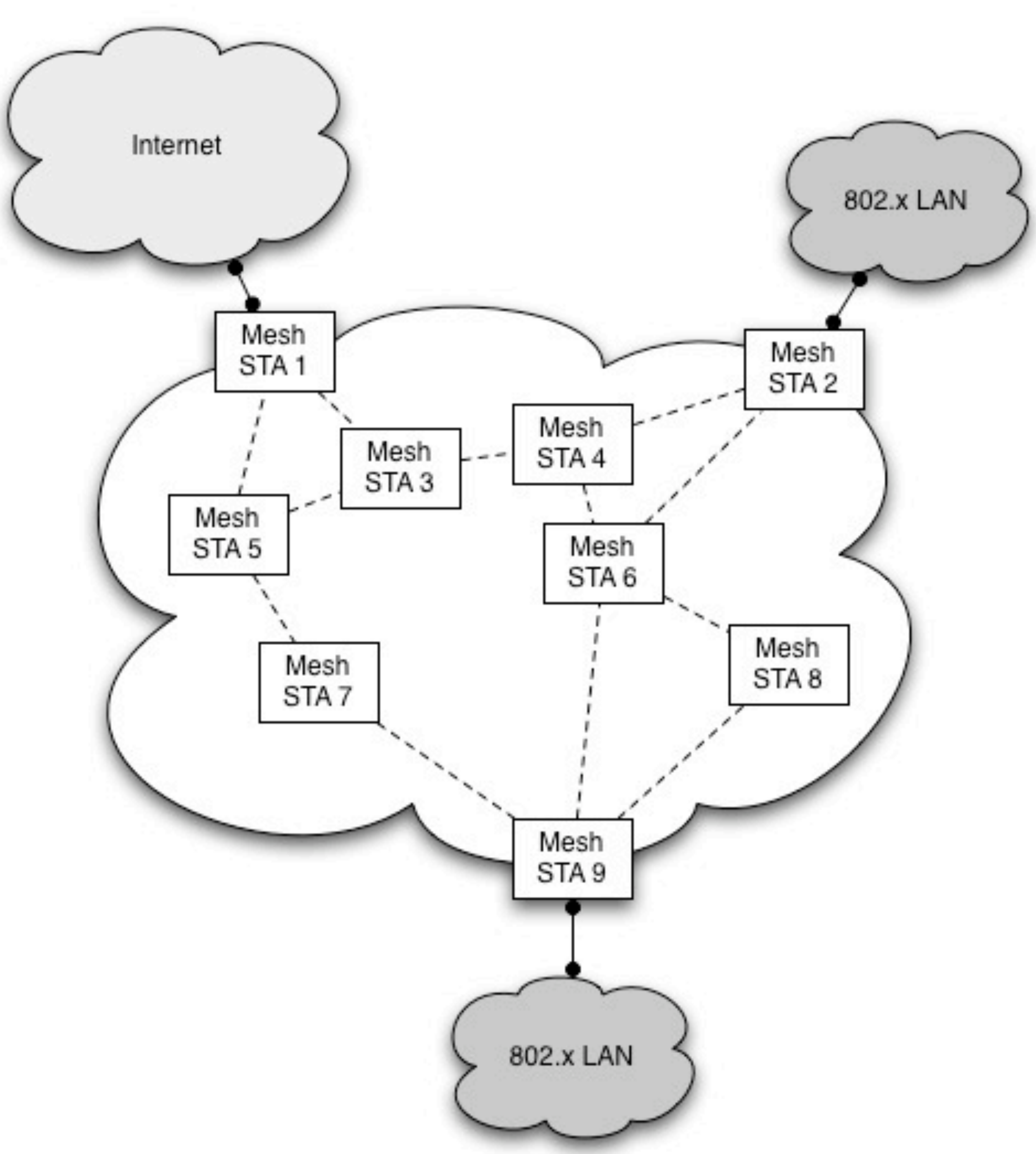
Wireless Mesh Networks under FreeBSD



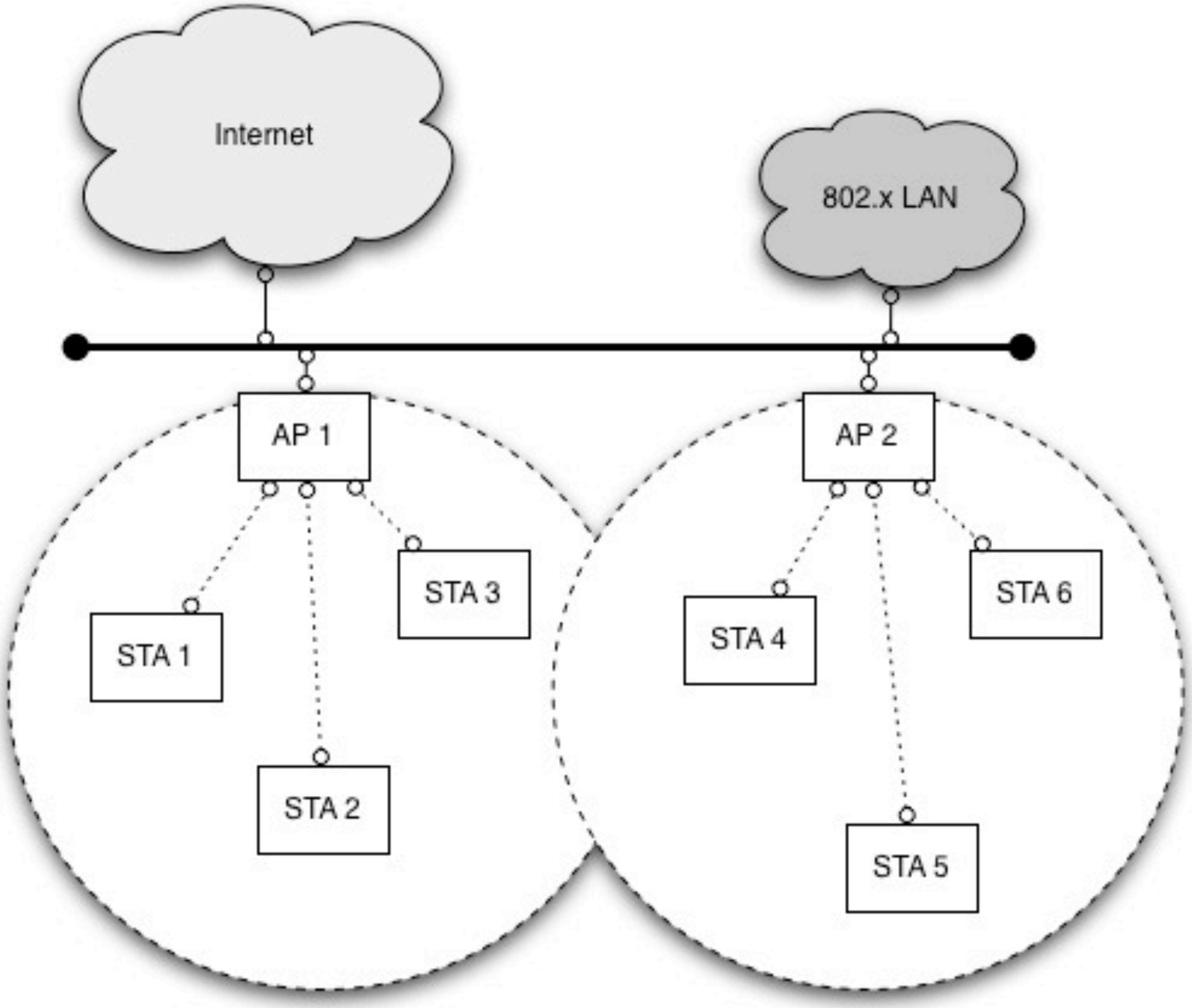
BSDCan 2010

Rui Paulo - rpaulo@FreeBSD.org

FreeBSD®

Typical wireless network

- Relatively easy to setup
- Pricey if you want to cover a wide area
- Bandwidth shared by all wireless STAs on the same BSS



What’s a wireless mesh network?

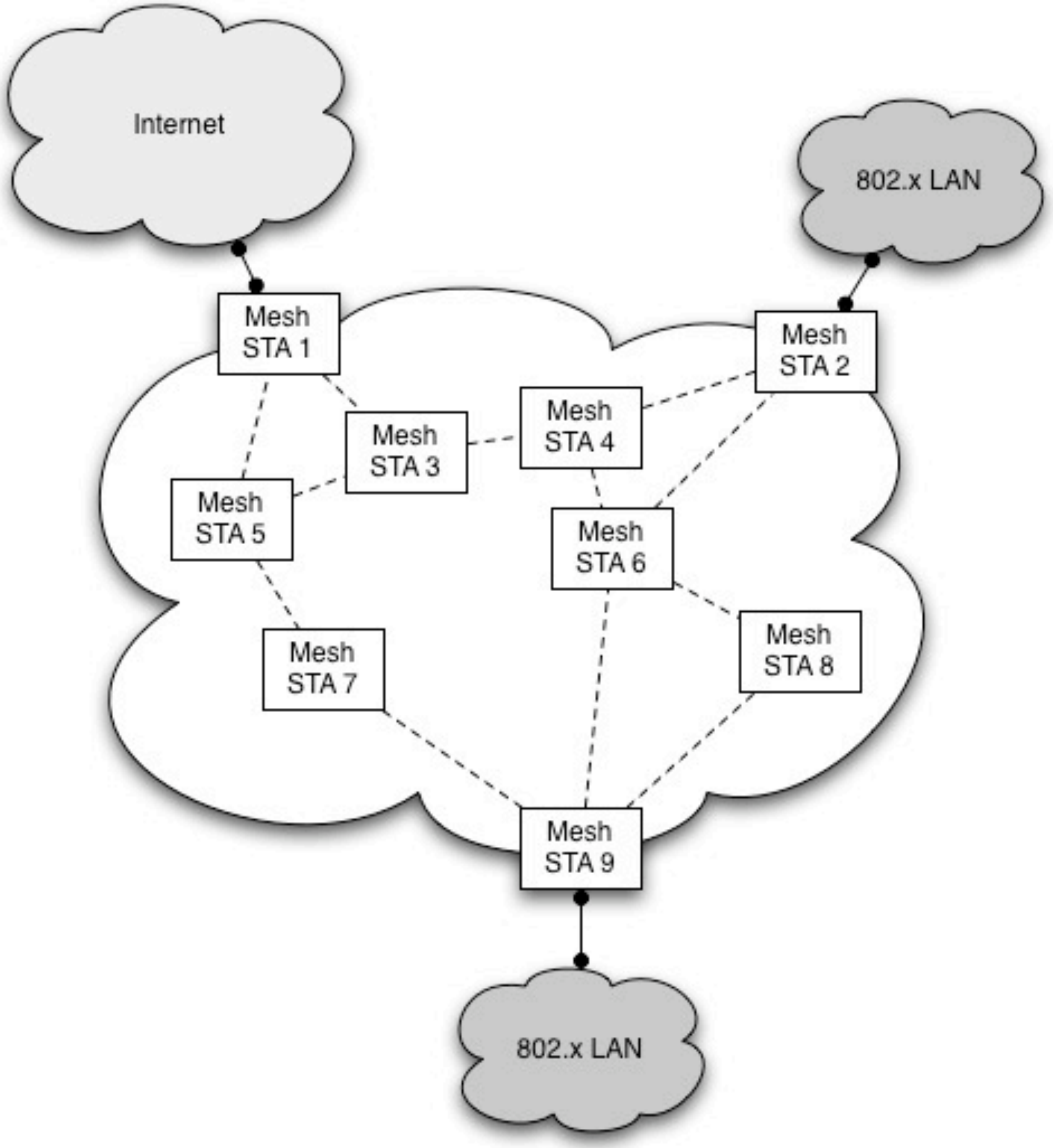
- Stations talk between each other (no central Access Point)
- Incorporates routing algorithms
- Local neighbors (peers) reachable with 0 hops
- Other nodes reachable with >= 1 hop(s)
- Several technologies available. We'll talk about 802.11s

Why use a mesh network?

- Self configuring solution to expand existing wireless network
- Low cost
- Complicated topologies, including no line-of-sight
- Implement a WAN/MAN on wireless with low cost

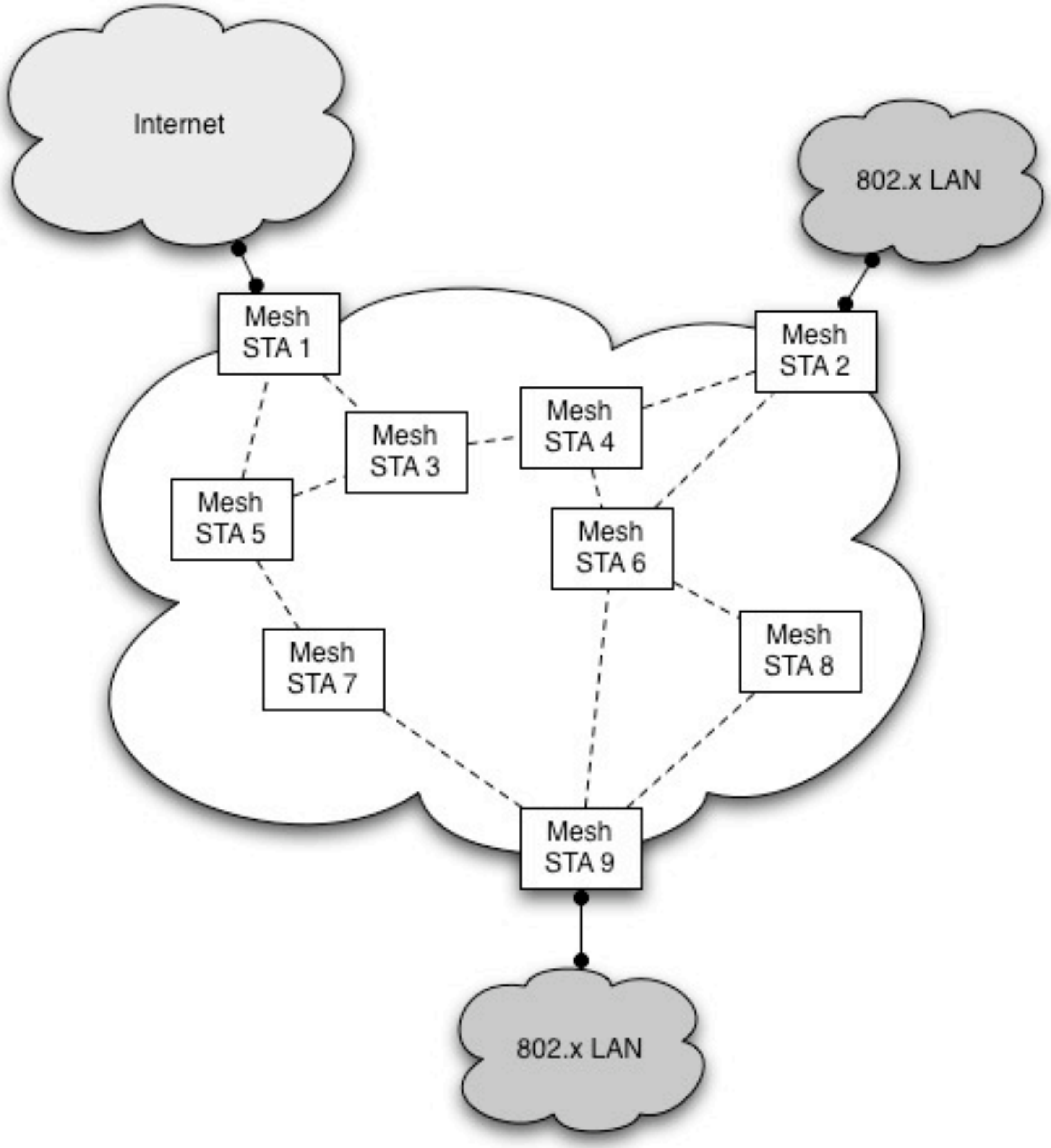
What does it look like?

- Group of Mesh STAs: MBSS
- Mesh peers of Mesh STA 1 are Mesh STA 3 and Mesh STA 5
- Mesh Portals (bridges) connect the mesh to the rest of the world



What does it look like?

- Note that 802.x LAN can be wired or wireless
- So we can combine Mesh + AP or Mesh + Wired
- The result is called ESS (Extended Service Set)



Examples of mesh networks

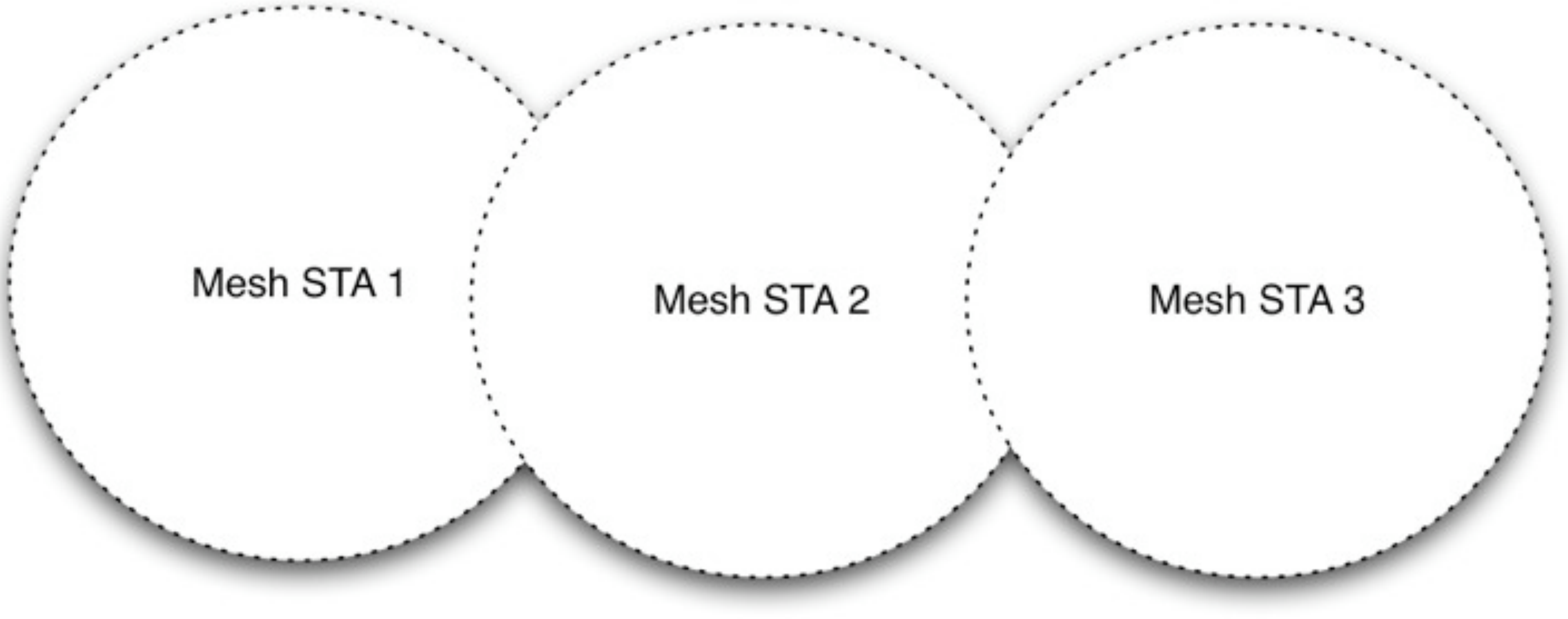
- Meraki Mesh (special long range radio)
- Mesh Dynamics (multiple radios)
- OLPC XO-1 children's laptop
- Smesh (fast roaming)
- SolarMesh (mesh STA power comes from solar energy)
- SONOS multi-room music system

Introducing 802.11s

- IEEE implementation of wireless mesh networks
- Amendment of 802.11-2007
- No changes on the 802.11 MAC header
- Currently under draft status – expected final version in <= 2015
- Most of the mesh configuration is optional – you can use your own routing algorithm

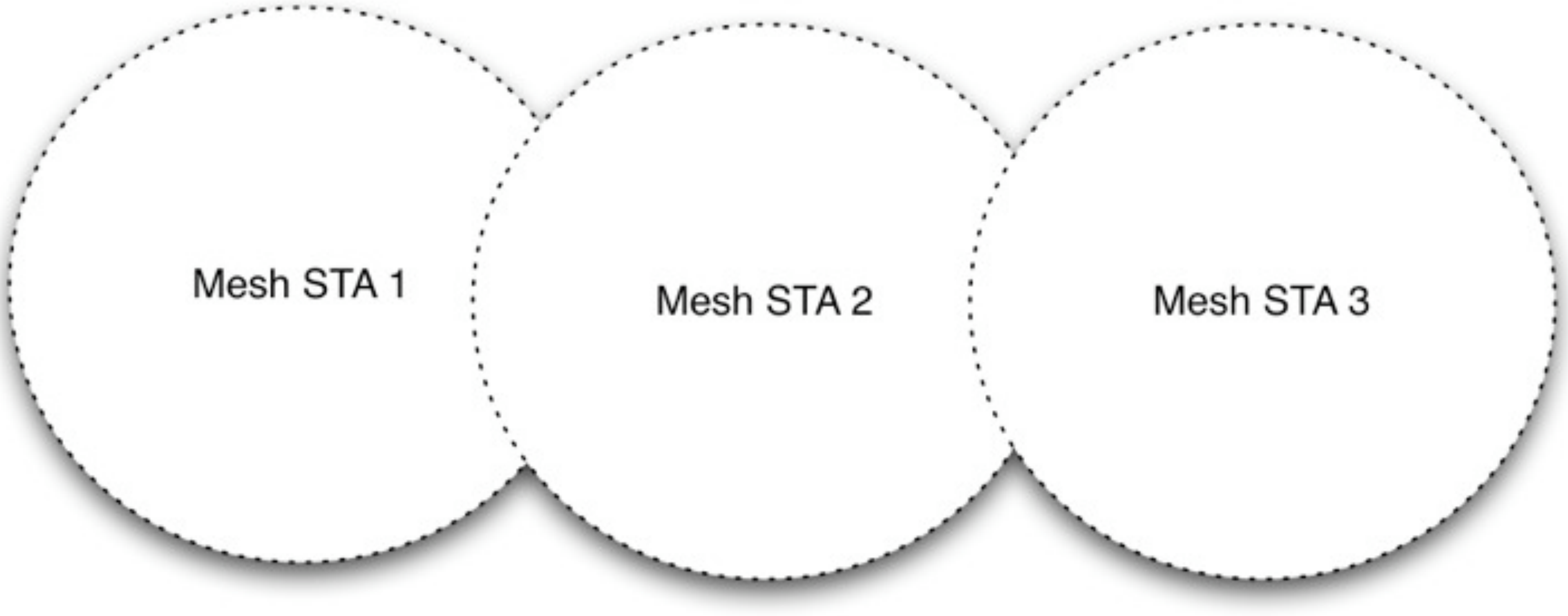
802.11s Peer discovery

- Mesh STA 1 peers with Mesh STA 2
- Mesh STA 3 peers with Mesh STA 2
- Mesh STA 2 peers with both

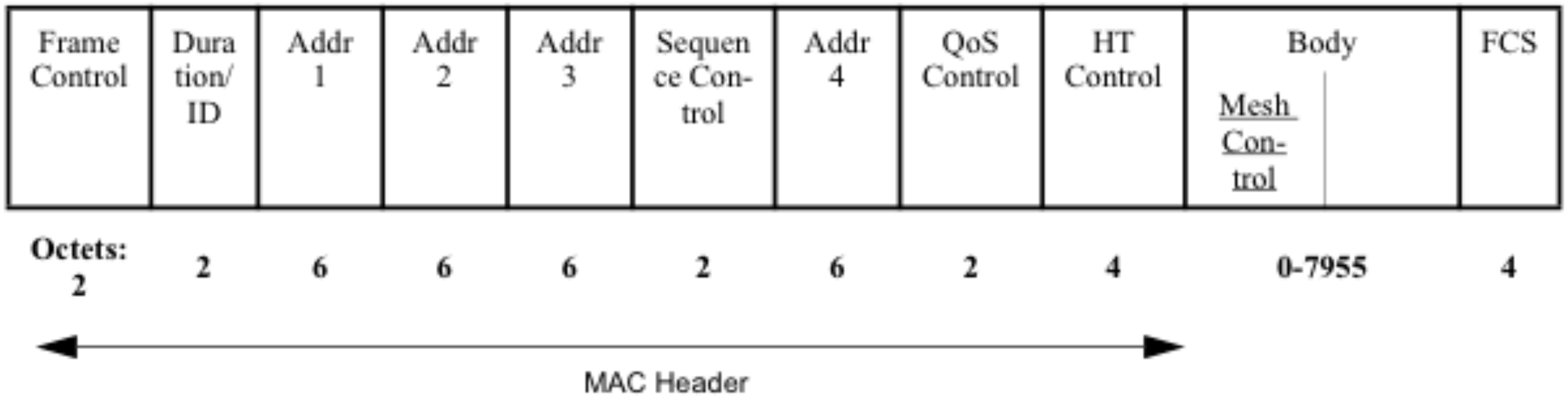


802.11s Peer discovery

- Mesh STA 1 and Mesh STA 3 can now talk to each other via Mesh STA 2
- You can use MAC ACLs to stop nodes from peering

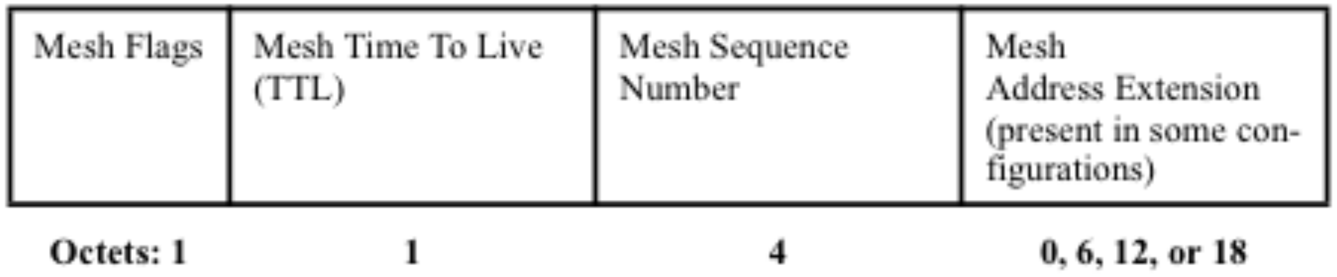


802.11s Mesh Header



Frame Control	Duration/ID	Addr 1	Addr 2	Addr 3	Sequence Control	Addr 4	QoS Control	HT Control	Body Mesh Control	FCS
Octets: 2	**2**	**6**	**6**	**6**	**2**	**6**	**2**	**4**	**0-7955**	**4**

MAC Header



Mesh Flags	Mesh Time To Live (TTL)	Mesh Sequence Number	Mesh Address Extension (present in some configurations)
Octets: 1	**1**	**4**	**0, 6, 12, or 18**

802.11s Algorithms

- Hybrid Wireless Mesh Protocol (HWMP) is the default routing algorithm (mandatory)
- Radio Aware Optmized Link State Routing (RA-OLSR) is the optional routing suggested by the spec
- FreeBSD implements HWMP for now – code is modular enough that it should be easy to support RA-OLSR

802.11s Algorithms

- Airtime is the default link metric algorithm (mandatory) – metric based on TX rate and error rate
- Authentication uses Simultaneous Authentication of Equals (SAE) (pre-shared secret)
- FreeBSD doesn't implement authentication, because we are waiting for the final standard

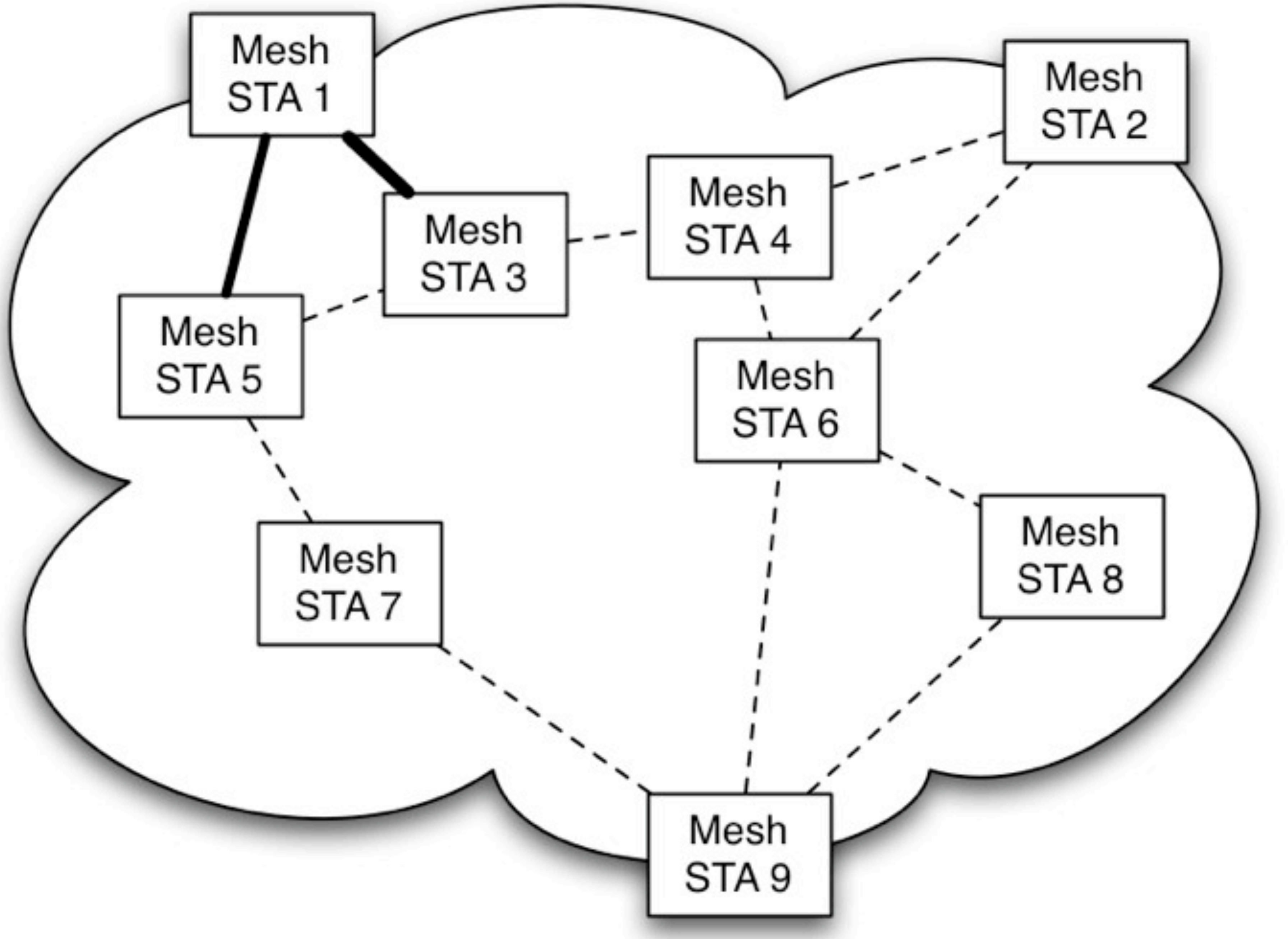
Hybrid Wireless Mesh Protocol

- Based on AODV (Ad-hoc On-Demand Distance Vector)
- On-Demand routing requires an exchange of path setup packets before actual data transmission
- Extended to enable proactive routing
- Proactive routing enables a root mesh STA to discover all nodes on the mesh

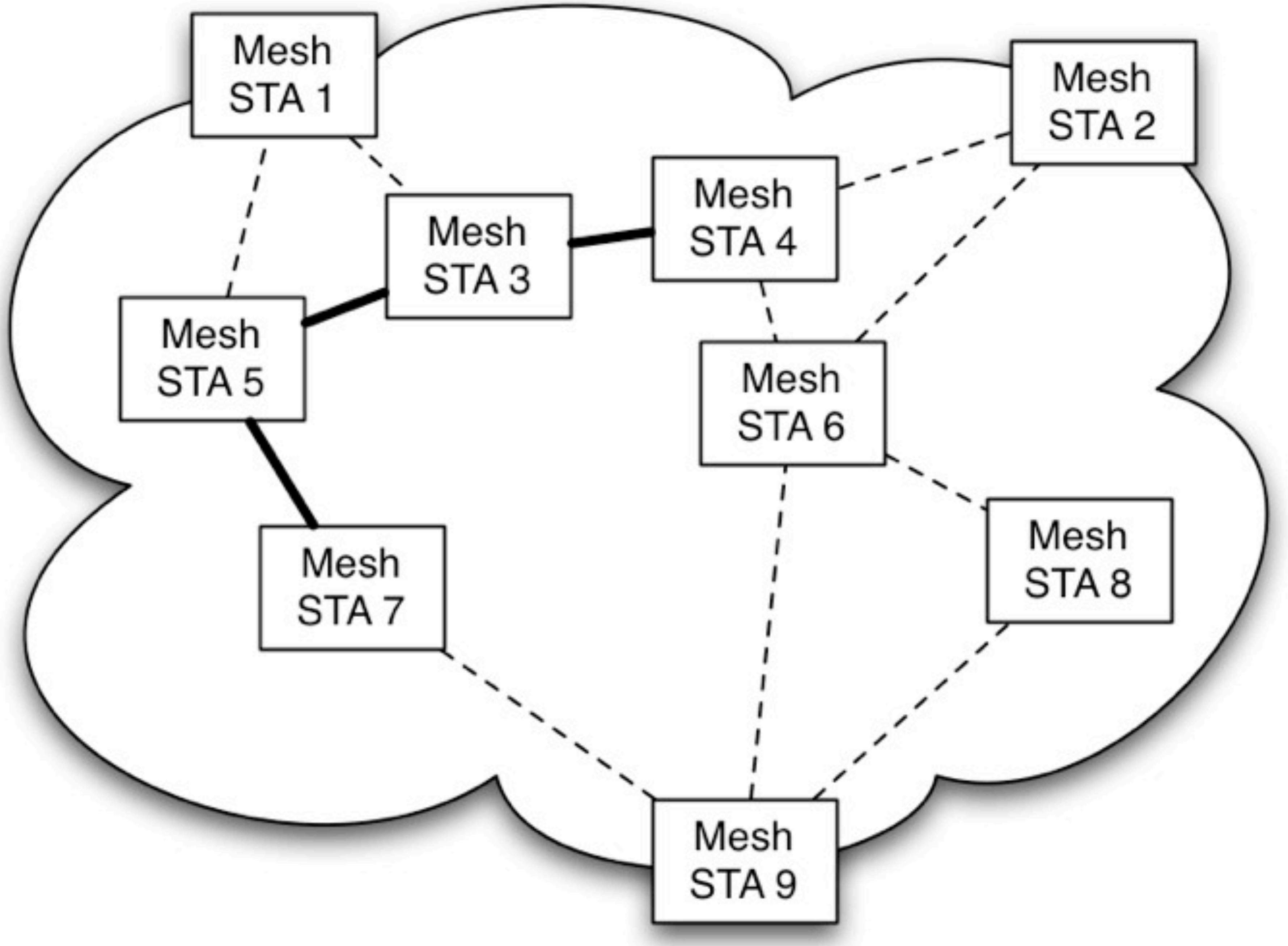
Hybrid Wireless Mesh Protocol

- Mesh STAs use the root mesh STA as a way to reach other mesh nodes faster
- “Hybrid” comes from on-demand + proactive
- Let's look at some of the common scenarios with HWMP

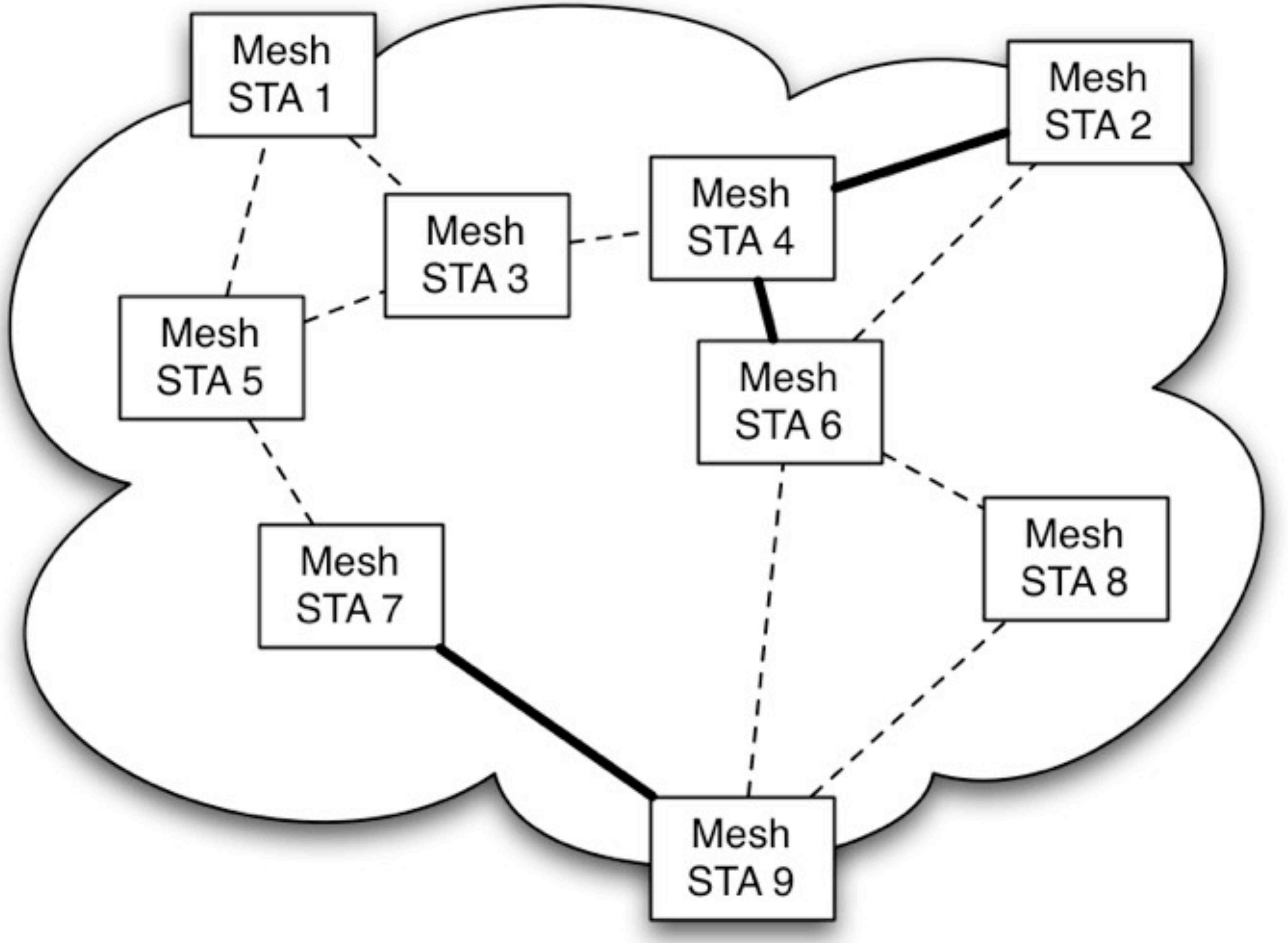
HWMP Path Request (STA 1 to STA 8)



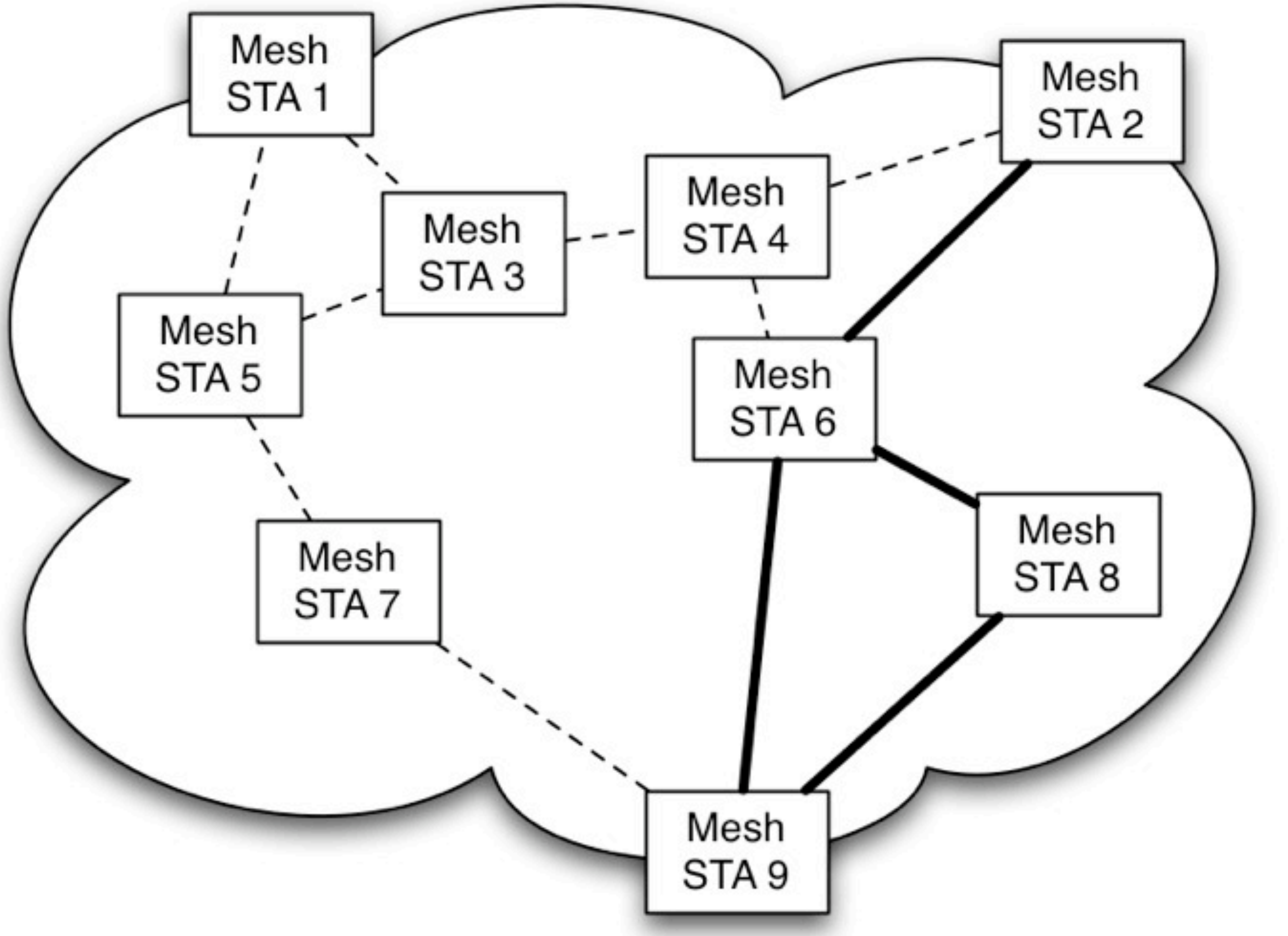
HWMP Path Request (STA 1 to STA 8)



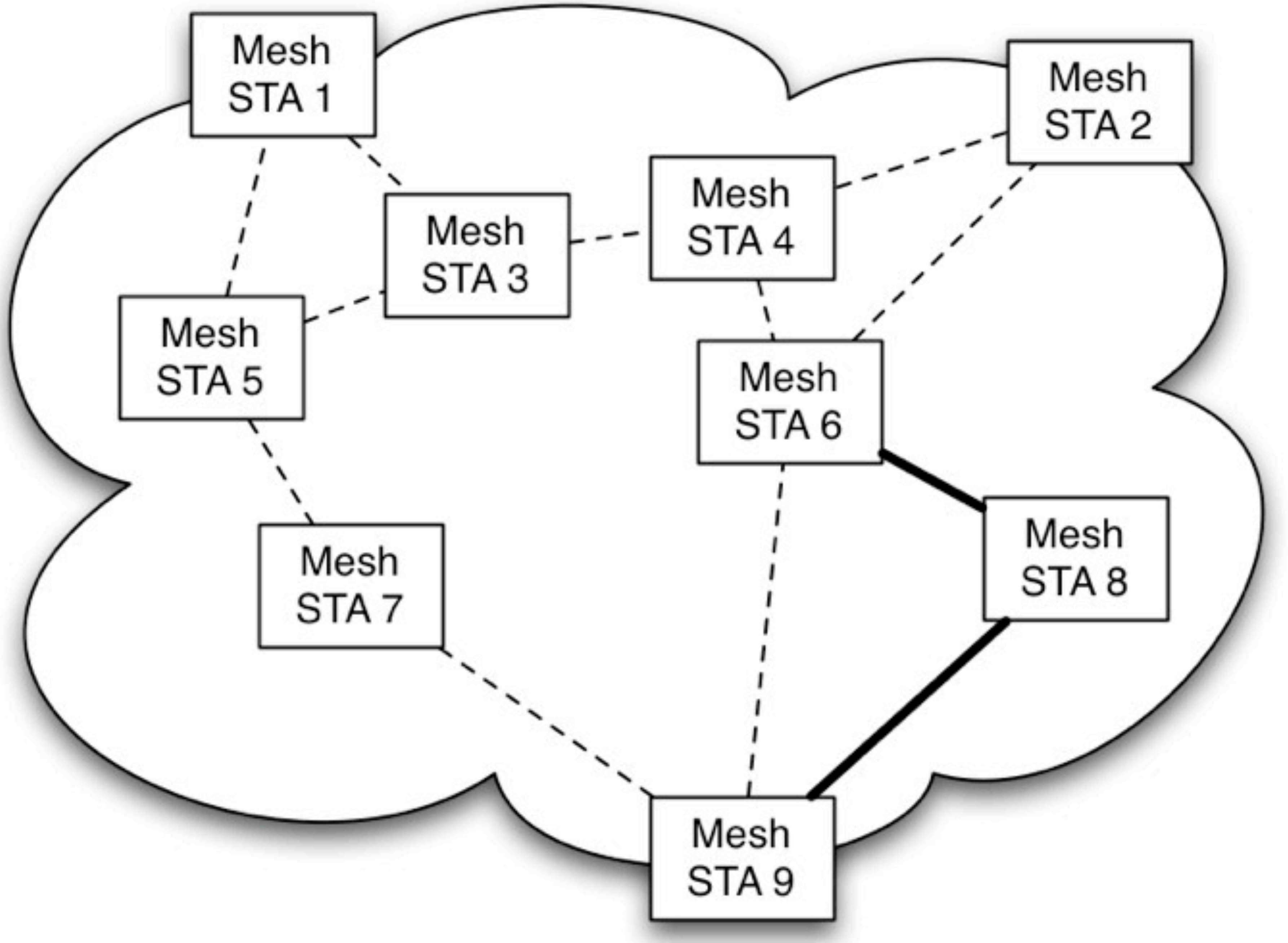
HWMP Path Request (STA 1 to STA 8)



HWMP Path Request (STA 1 to STA 8)

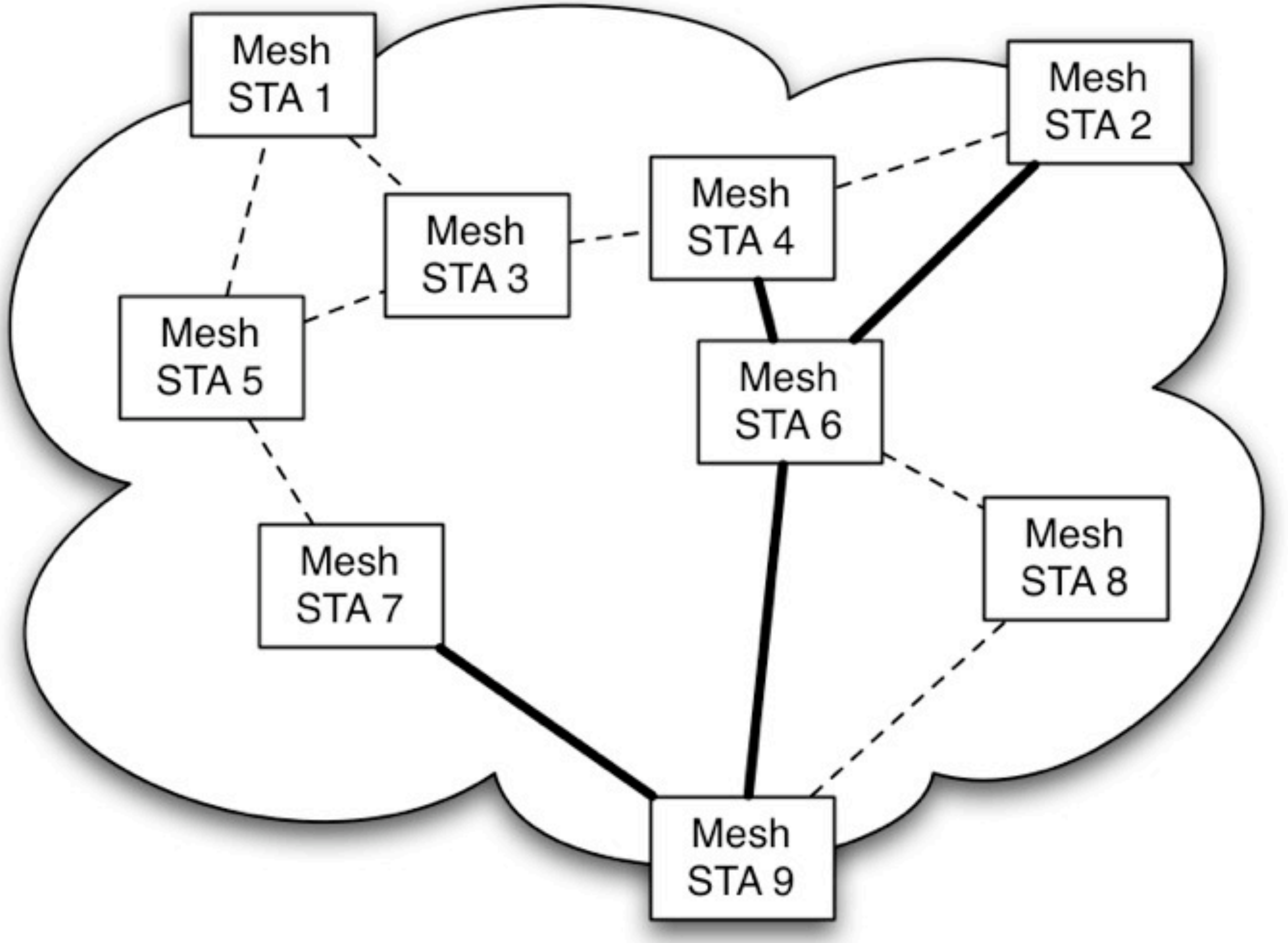


HWMP Path Reply (STA 8 to STA 1)

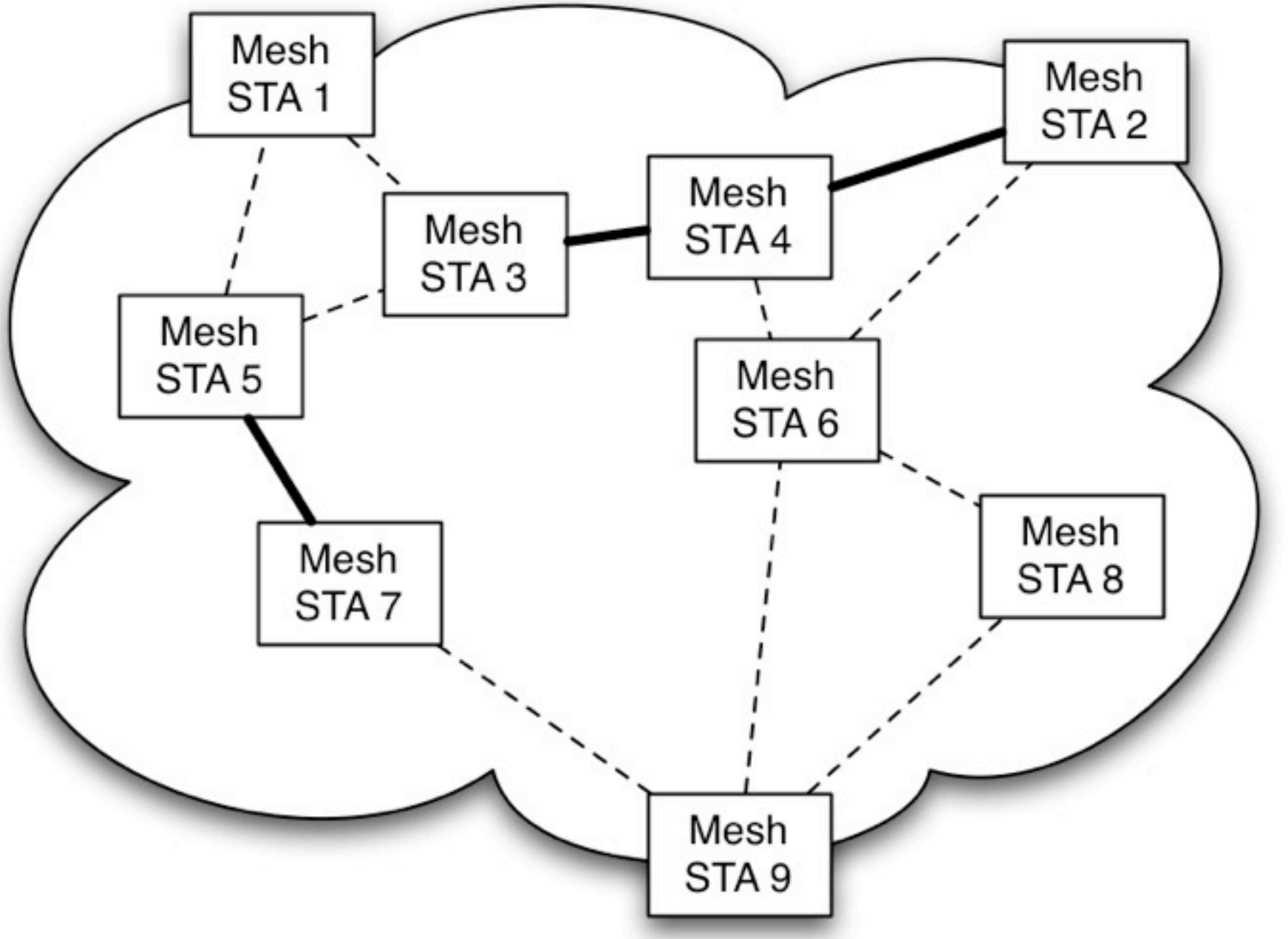


FreeBSD®

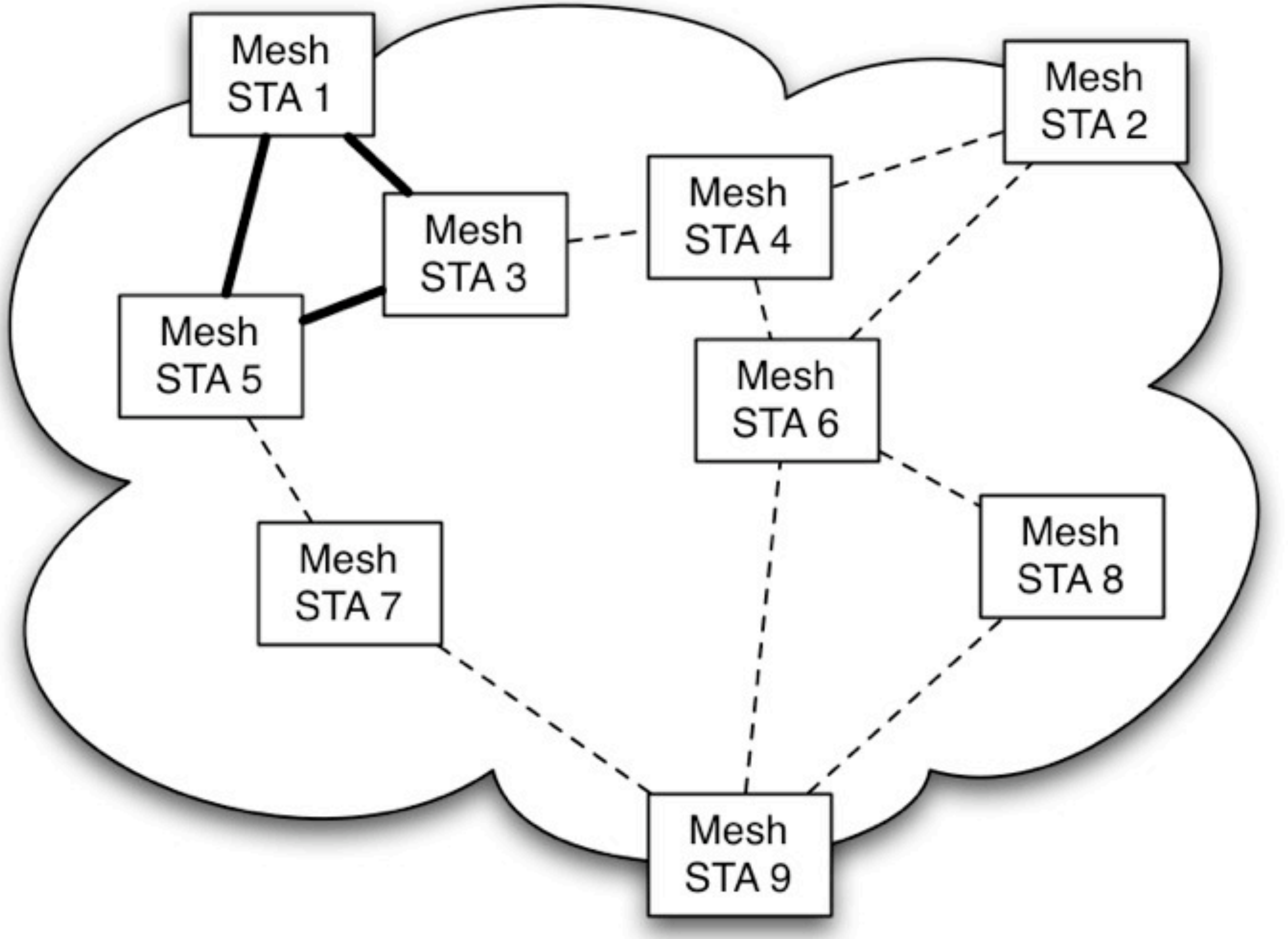
HWMP Path Reply (STA 8 to STA 1)



HWMP Path Reply (STA 8 to STA 1)

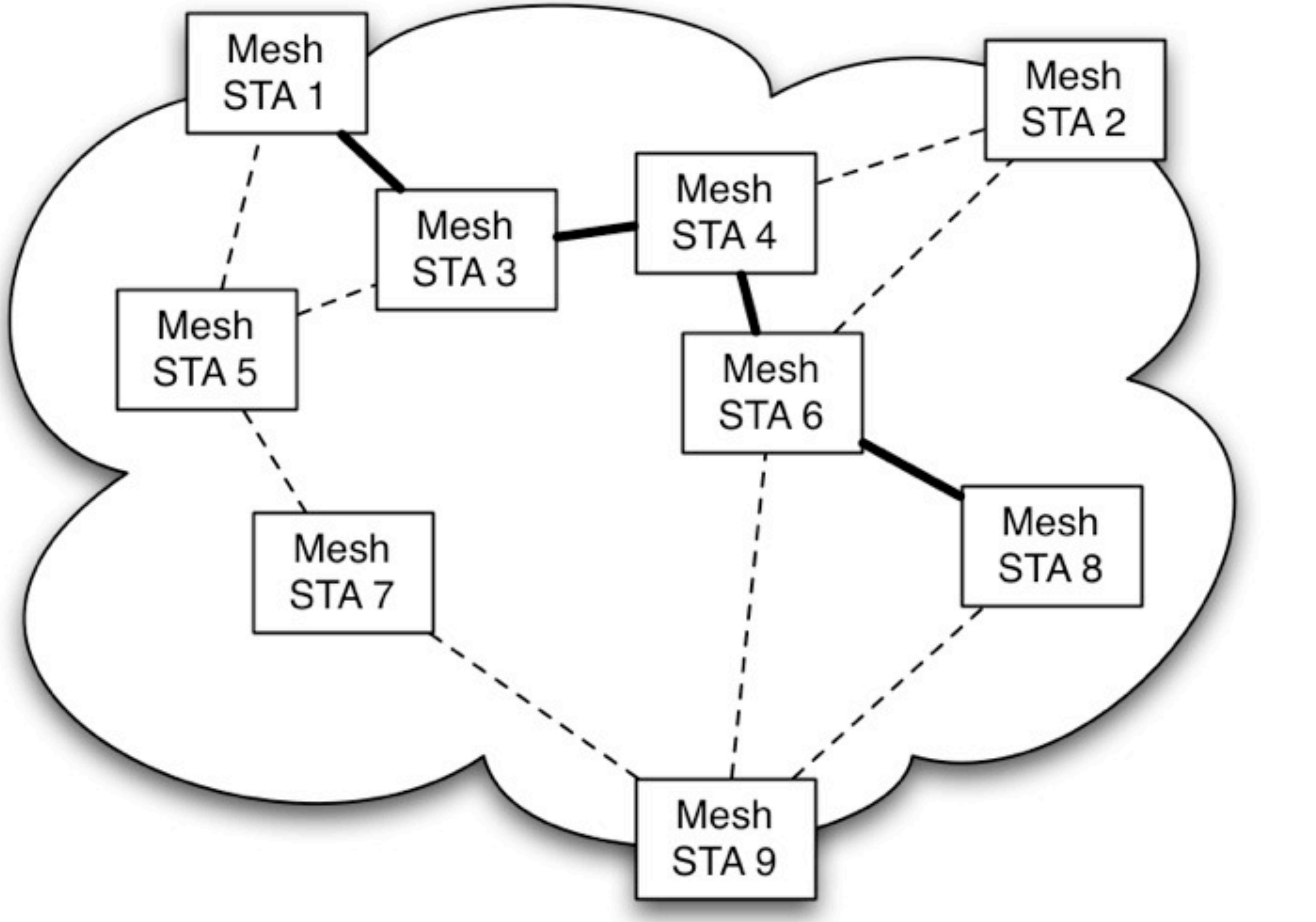


HWMP Path Reply (STA 8 to STA 1)

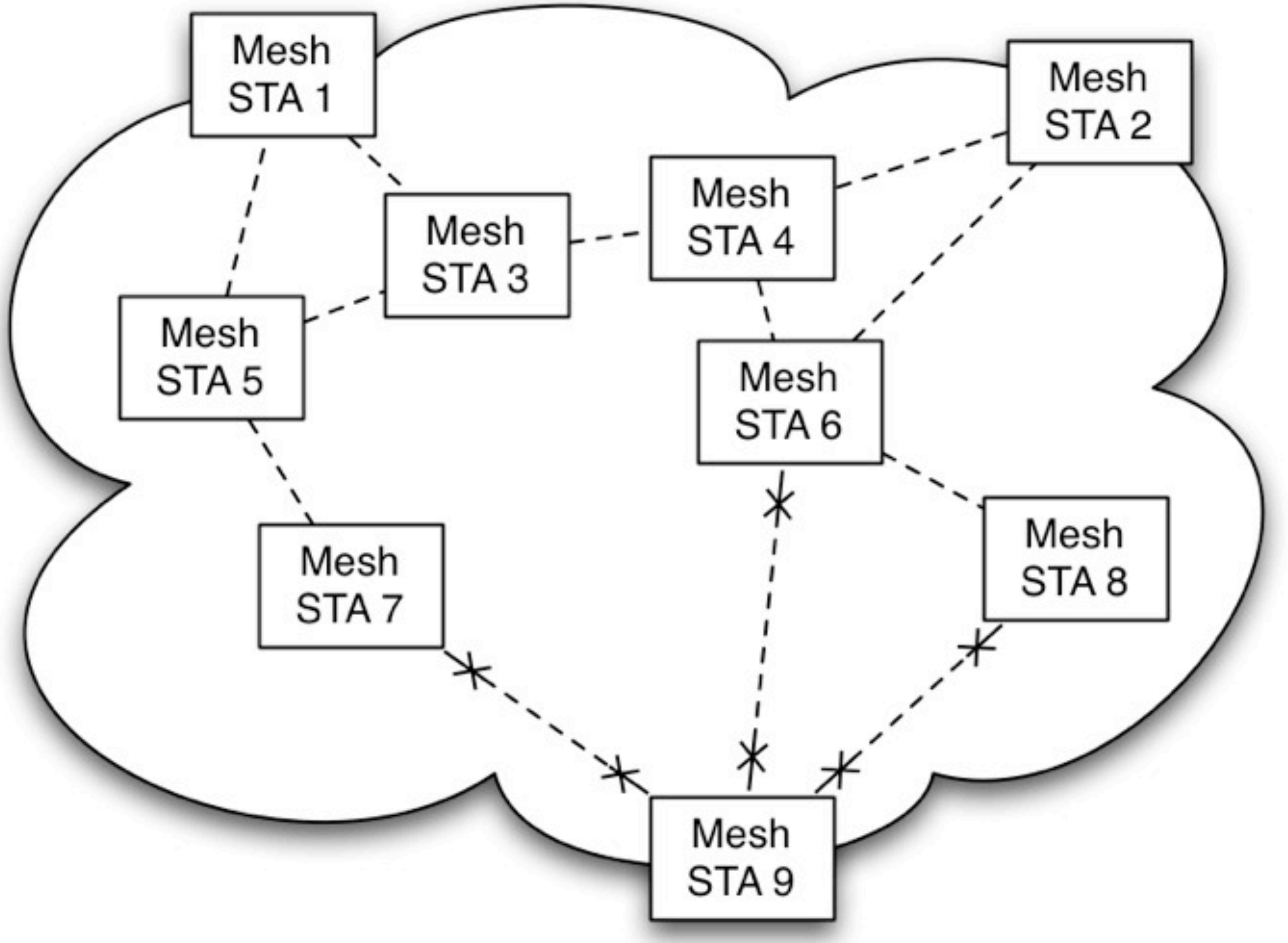


FreeBSD

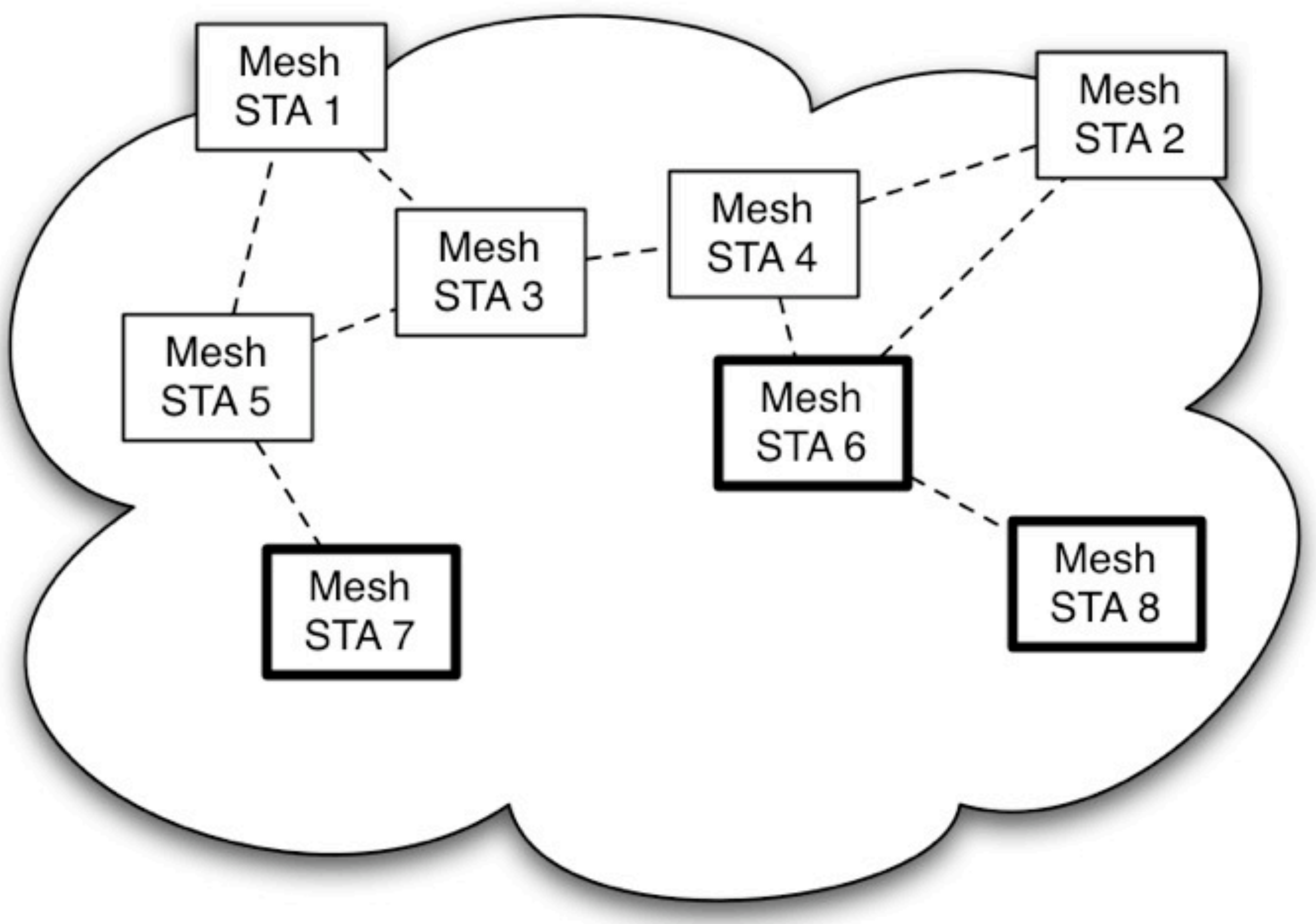
HWMP Path: STA 1 to STA 8



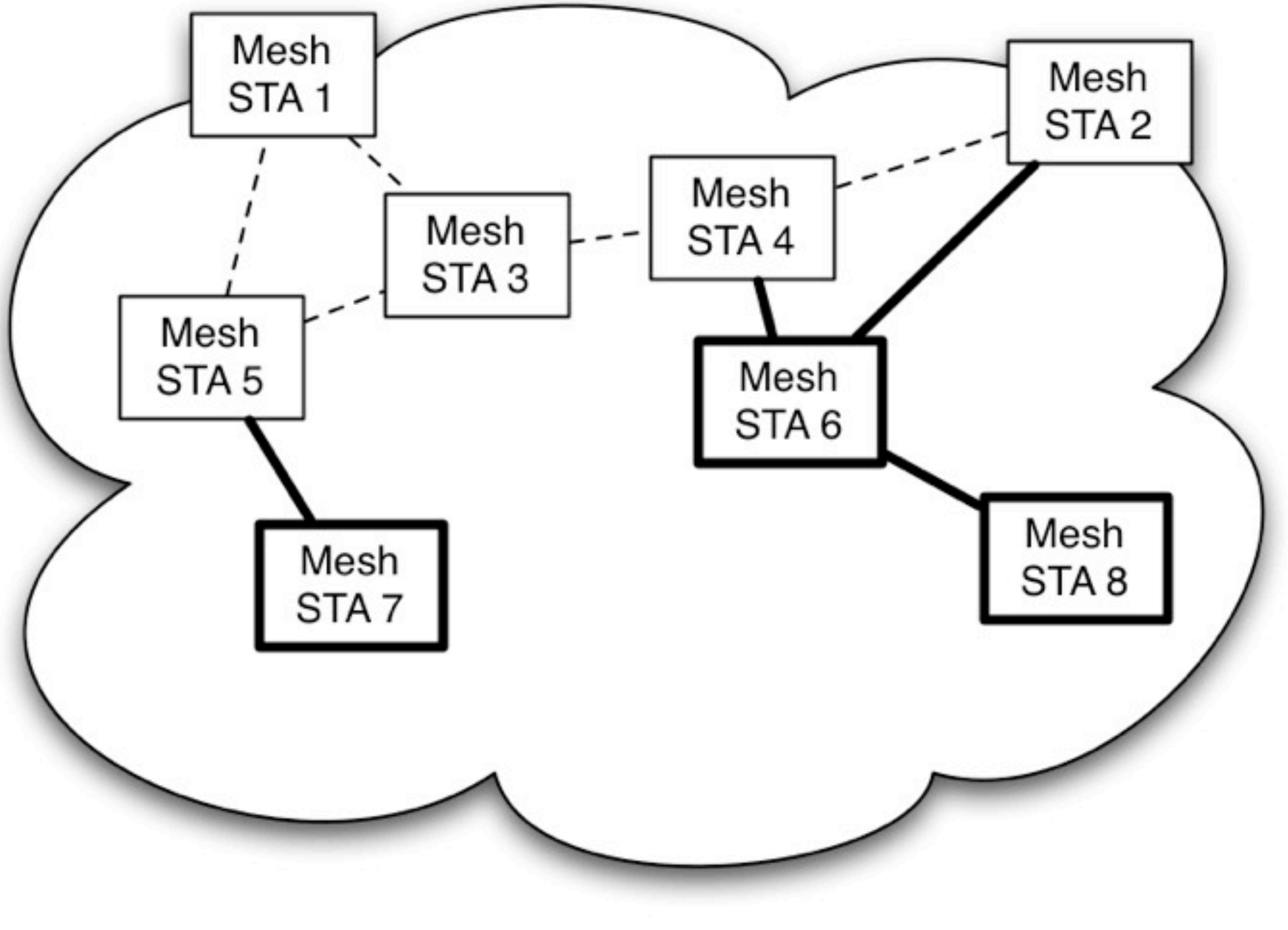
HWMP Topology Change



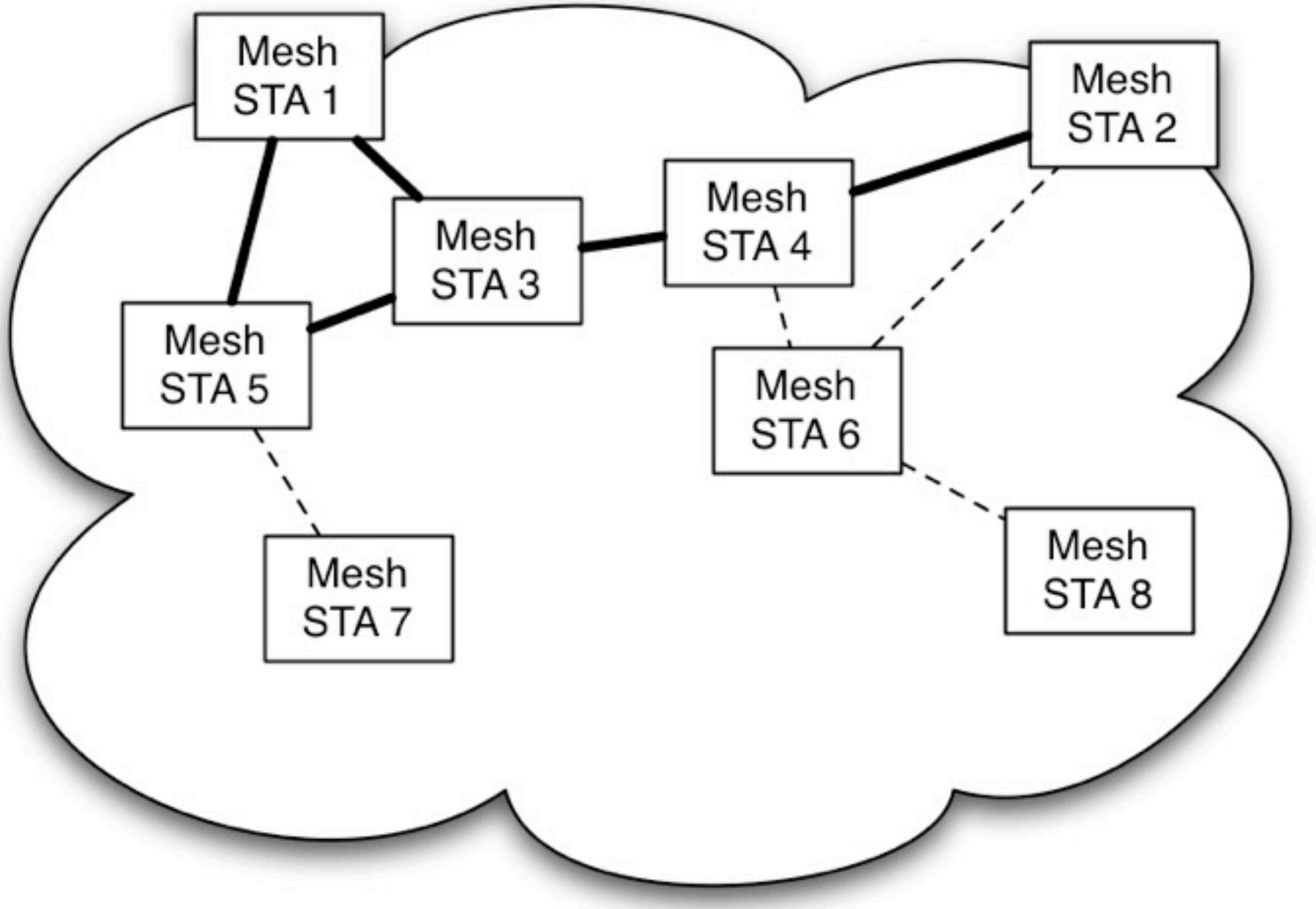
HWMP Topology Change



HWMP Topology Change



HWMP Topology Change



Notes on HWMP

- Sequence numbers are used to drop old packets & to avoid loops
- Each station tracks the last seq. number seen from the other stations
- Path setup may take a while (seconds) if the number of hops is high
- Packets must be queued while node discovery happens

802.11s on net80211

- Second public implementation of 802.11s
- Experimental status
- First release in FreeBSD 8.0
- Sponsored by The FreeBSD Foundation
- Started on late April
- Working implementation on late July
- Linux compatible

802.11s on net80211

- Each wlan driver needs to be changed for mesh support
- Drivers working already: ath(4), ral(4) and mwl(4)
- Firmware-based drivers (like ipw, iwi, wpi, etc.) won't work
- Drivers that do hostap, can be changed to work with mesh mode easily
- Next step is to play with USB wlan drivers

802.11s on net80211: user side

- ifconfig wlan0 create wlanmode mesh channel <chan> meshid freebsd-mesh

```
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:0b:6b:2d:dc:d8
      media: IEEE 802.11 Wireless Ethernet autoselect mode 11a <mesh>
      status: running
      meshid freebsd-mesh channel 36 (5180 Mhz 11a) bssid 00:0b:6b:2d:dc:d8
      regdomain ETSI country PT ecm authmode OPEN privacy OFF txpower 17
      mcastrate 6 mgmtrate 6 scanvalid 60 wme burst bintval 1000 meshttl 31
      meshpeering meshforward meshmetric AIRTIME meshpath HWMP
      hwmprootmode DISABLED hwmpmaxhops 31
```

802.11s on net80211: user side

- `ifconfig wlan0 list sta`

```
ADDR              CHAN LOCAL  PEER   STATE RATE RSSI IDLE  TXSEQ  RXSEQ
00:0b:6b:2d:dc:d8   36     0     0    IDLE   0M 18.5   15      1    192
00:0b:6b:2d:db:ac   36  9827  a5b3   ESTAB  54M 14.0    0      2  28752
00:0b:6b:2d:dd:17   36  afdb  ab30   ESTAB  54M 19.0    0      5  25024
00:0b:6b:87:1c:f0   36  1904  825c   ESTAB  54M  6.0    0     30    192
```

- `ifconfig wlan0 list mesh`

```
DEST              NEXT HOP          HOPS METRIC LIFETIME   MSEQ FLAGS
00:0b:6b:2d:dd:17 00:0b:6b:2d:dc:d8    1   2842     5000      9    V
00:0b:6b:2d:dc:d8 00:0b:6b:2d:dc:d8    0      0     5000      0    V
00:0b:6b:2d:db:ac 00:0b:6b:2d:dc:d8    1    347     5000      4    V
```

802.11s on net80211: user side

- `ifconfig wlan0 hwmprootmode NORMAL`
  - Root mesh station discovers nodes using PREQ packets.
- `ifconfig wlan0 hwmprootmode PROACTIVE`
  - Root mesh station discovers nodes and asks for proactive PREPs. This means that the mesh STA will always send a PREP even if it already has a path to the root mesh STA.
- `ifconfig wlan0 hwmprootmode RANN`
  - Root mesh station discovers nodes using RANN packets.

Performance measurements

- For 0 hops performance is the same as adhoc mode
- As hops increase, performance decreases about 50%
- E.g.: 0 hop: ~28Mbps / 1 hop: ~14Mbps / 2 hops: ~7Mbps / 3 hops: ~3.5Mbps / etc.
- You can also use a 802.11n card which makes the mesh a bit faster

Performance measurements

- Performance can be increased by use of Mesh Coordinated Channel Access (MCCA)
- MCCA works a bit like TDMA
- Mesh STA reserves a time slot and coordinates that time slot with all the neighbors
- Not yet implemented

Acknowledgments

- The FreeBSD Foundation for sponsoring this project - www.freebsdfoundation.org
- Sam Leffler for his patience, time and help
- Gateworks Corp. for a GW2358 ARM board www.gateworks.com
- Cozybit for the Wireshark patches enabling mesh sniffing - www.cozybit.com
- Your donations to the FreeBSD Foundation made this project possible!

References + Q&A

- http://people.freebsd.org/~rpaulo/asiabsdcon10-mesh.pdf
- http://wiki.freebsd.org/WifiMesh
- http://www.ieee802.org/11/Reports/tgs_update.htm
- http://en.wikipedia.org/wiki/IEEE_802.11s
- http://o11s.org/ - Linux implementation