# Netflix and FreeBSD: Observations from ~3 Years of Running Head

Jonathan Looney
BSDCan 2019

# Open Connect

NETFLIX AND FREEBSD
BSDCan 2019

**Open Connect** is Netflix's CDN. It is **global**, **efficient**, and **purpose-built** for distributing Netflix's content.

# The Open Connect Appliance

The OCA is the workhorse of the Open Connect network.

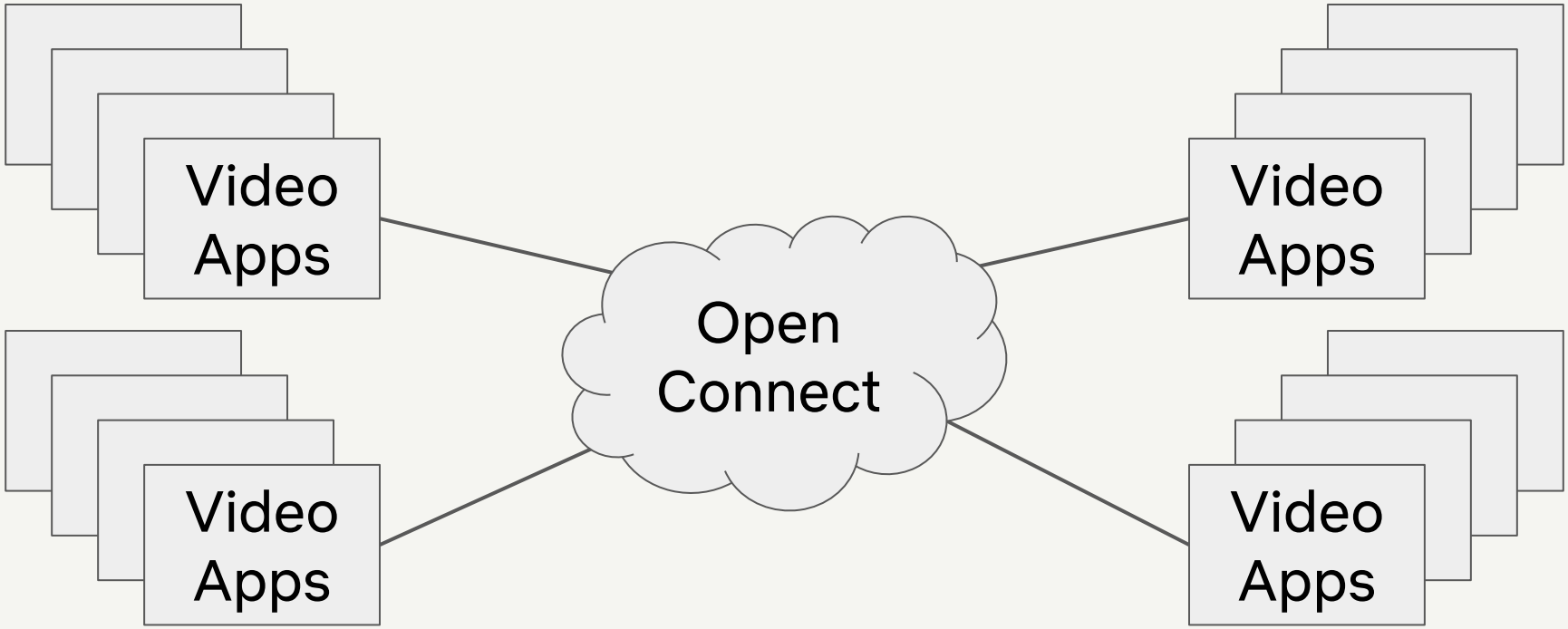
The OCA almost exclusively runs open-source software.


40Gb/s Storage Appliance with 248TB storage (2RU form factor)



NETFLIX

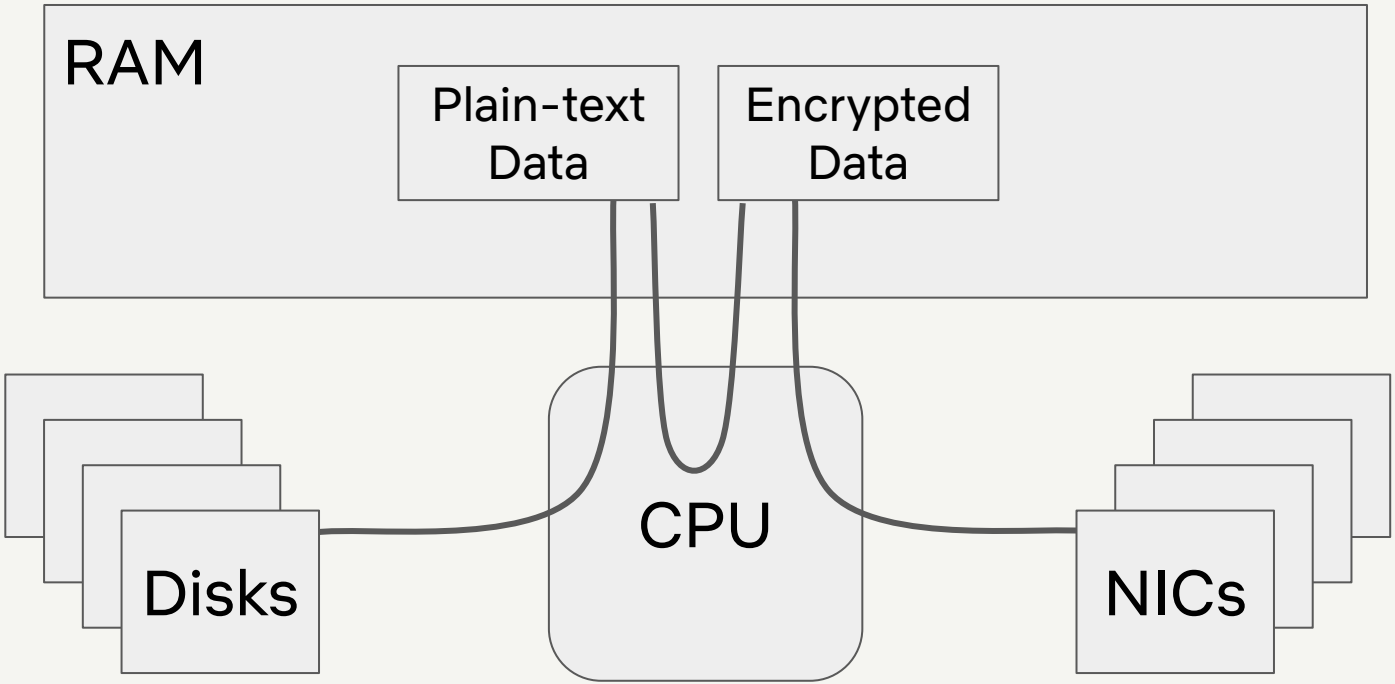
# Open Connect Traffic






NETFLIX

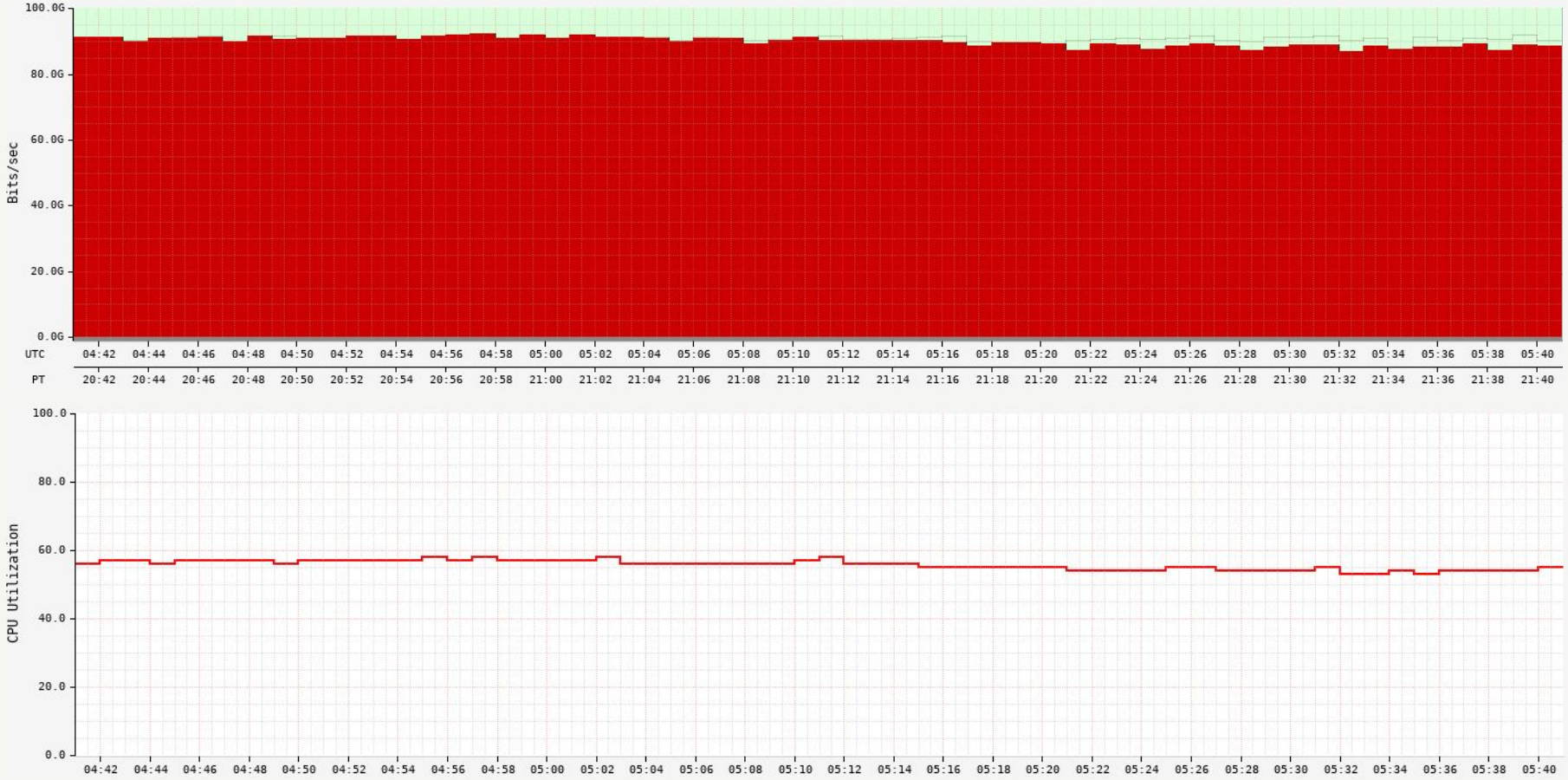
# Netflix OCA Workload






NETFLIX

Using FreeBSD and commodity parts, we achieve 90 Gb/s serving TLS-encrypted connections with ~55% CPU on a 16-core 2.6-GHz CPU.


Bits/sec
100.0G
80.0G
60.0G
40.0G
20.0G
0.0G
UTC
PT
04:42 04:44 04:46 04:48 04:50 04:52 04:54 04:56 04:58 05:00 05:02 05:04 05:06 05:08 05:10 05:12 05:14 05:16 05:18 05:20 05:22 05:24 05:26 05:28 05:30 05:32 05:34 05:36 05:38 05:40
20:42 20:44 20:46 20:48 20:50 20:52 20:54 20:56 20:58 21:00 21:02 21:04 21:06 21:08 21:10 21:12 21:14 21:16 21:18 21:20 21:22 21:24 21:26 21:28 21:30 21:32 21:34 21:36 21:38 21:40
CPU Utilization
100.0
80.0
60.0
40.0
20.0
0.0
04:42 04:44 04:46 04:48 04:50 04:52 04:54 04:56 04:58 05:00 05:02 05:04 05:06 05:08 05:10 05:12 05:14 05:16 05:18 05:20 05:22 05:24 05:26 05:28 05:30 05:32 05:34 05:36 05:38 05:40

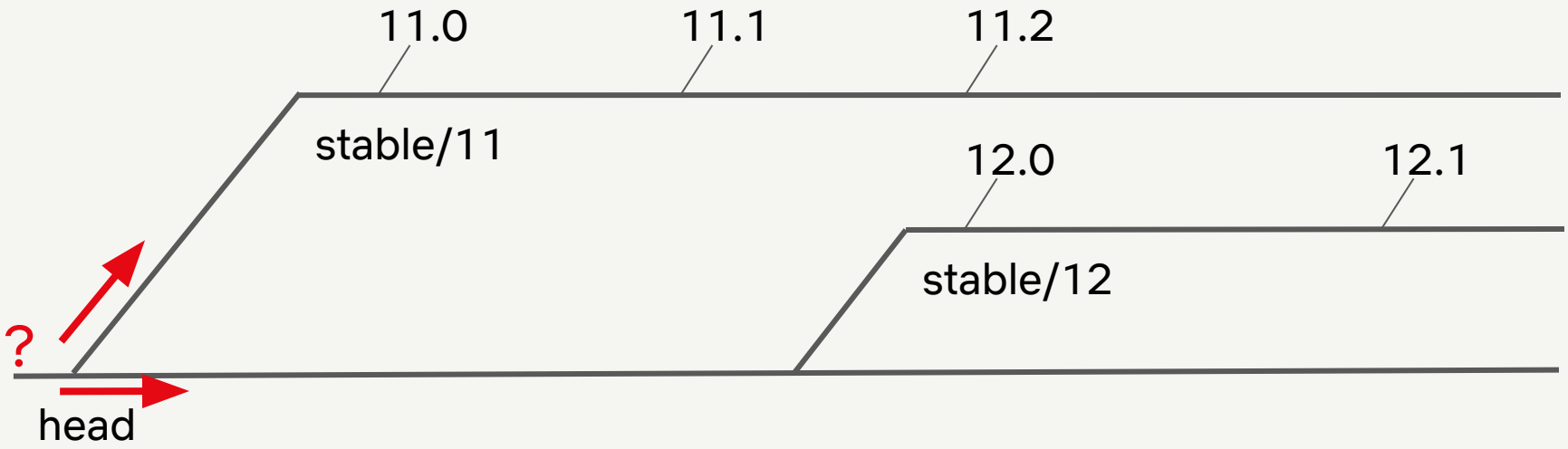
# OCA Operating System

NETFLIX AND FREEBSD
BSDCan 2019

# Why do we use FreeBSD?

# We came for the license. We stay for the efficiency.

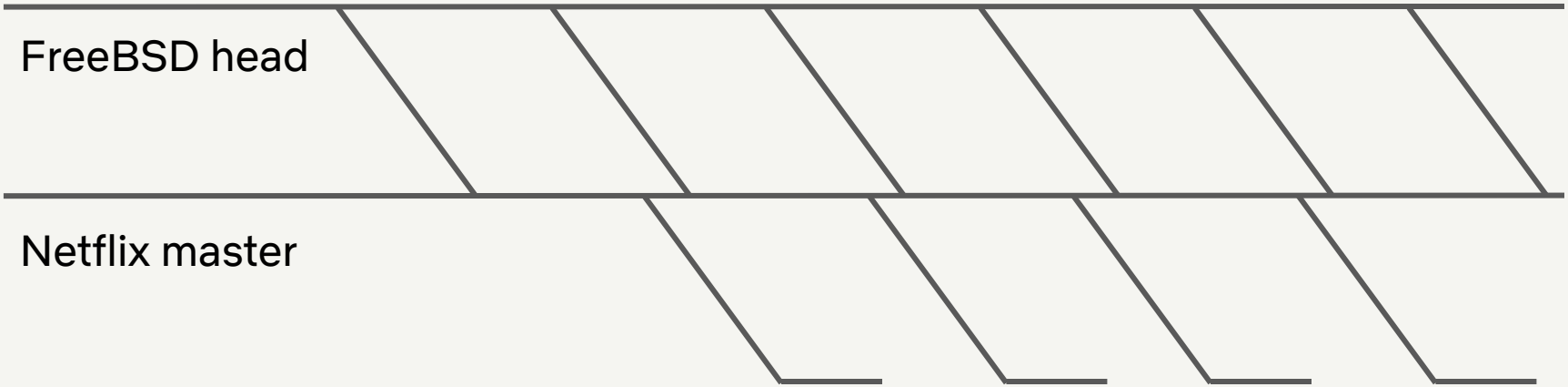
# FreeBSD Release Cycle






NETFLIX

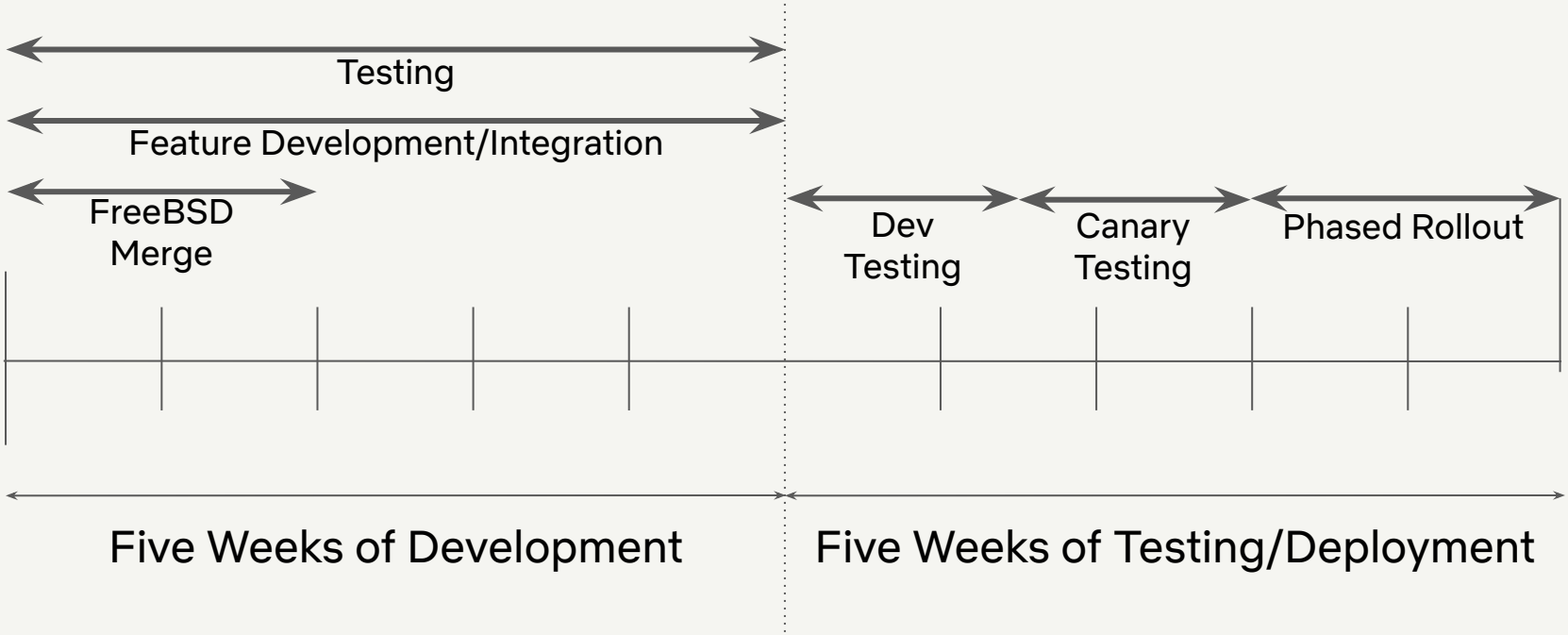
# We Track FreeBSD “Head”






NETFLIX

# Typical Release Cycle



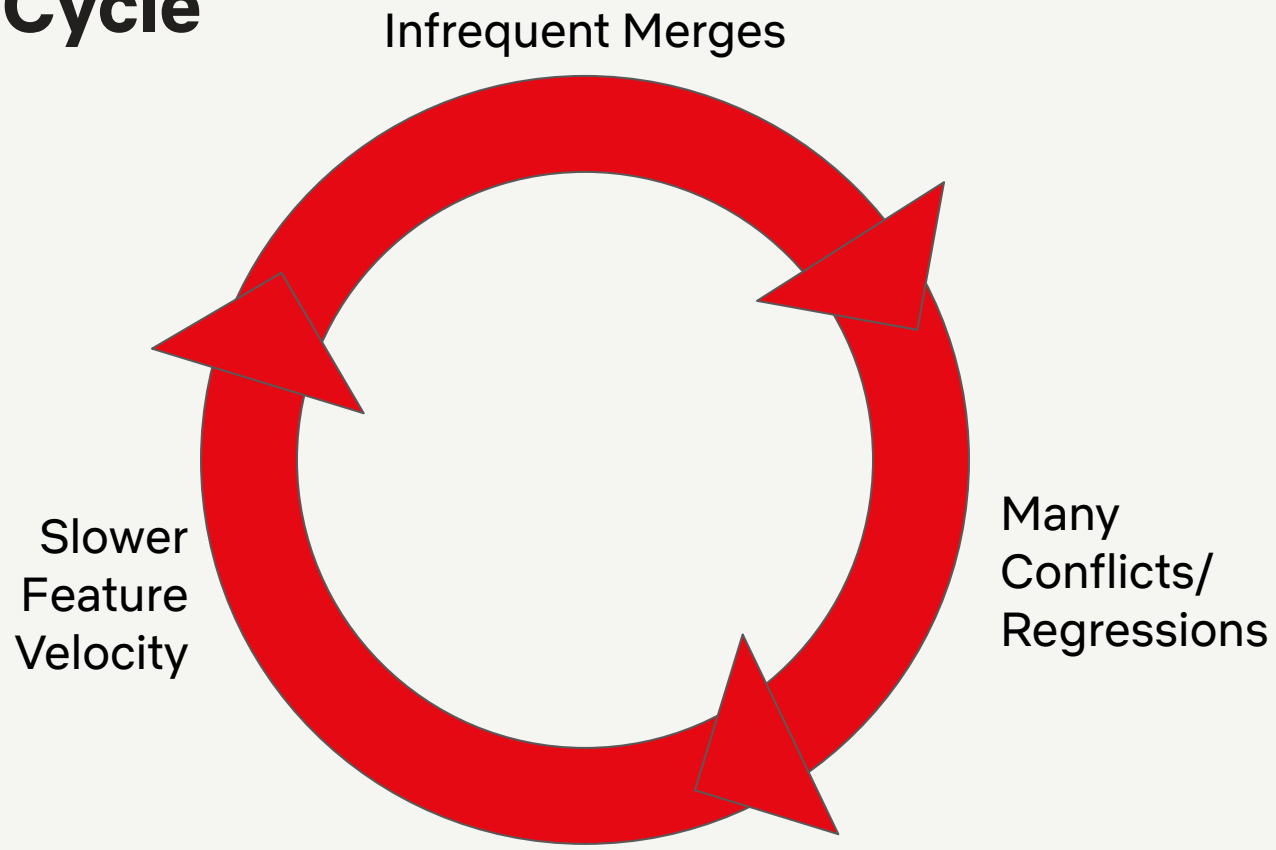
# Examples of Features

- NUMA enhancements
- Asynchronous sendfile
- Kernel TLS
- Pbuf allocation enhancements
- “Unmapped” mbufs
- I/O scheduling
- TCP algorithms
- TCP logging infrastructure

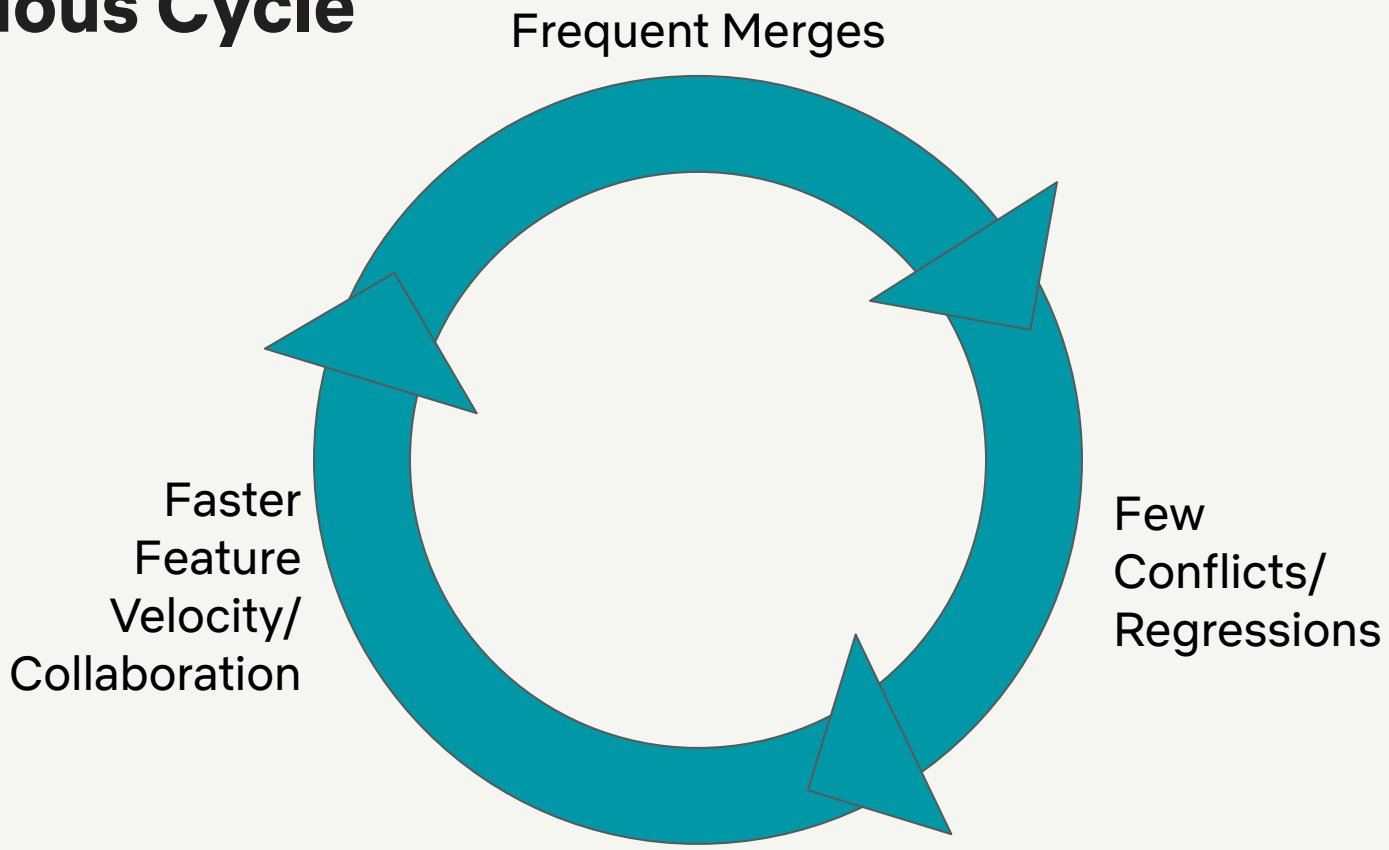
Tracking “head” lets us stay **forward looking** and focused on **innovation**.

# Downstream users of open-source projects can be stuck in “vicious” or “virtuous” cycles.

# Vicious Cycle



# Virtuous Cycle



# Reasons We Keep Local Diffs

- Information covered under NDA
- Feature which is still in development/testing
- Feature which needs to be generalized

# It is our intention to upstream any code which we can.

# Observations from Running Head

- Easier to collaborate with others
- Faster bug fixes and features
- Easier to upstream code
  - Also better, as what we upstream is the same code we run internally
- When tracking head, upstream code freezes are more disruptive than helpful
- API/KPI changes are easy to handle
- ABI/KBI changes are (mostly) a non-issue
- Head quality is so high that bug fallout is manageable

# Benefits to the FreeBSD Projects

- Wide deployment of “head” branch code (albeit in a narrow use case)
- Early intensive testing
- Incentive for Netflix to upstream code

# Objections to Running “Development” Code

- It isn’t stable
- Why should you pay to find the bugs others will find while testing head?
- Aren’t there more security bugs?
- No one runs development branches
- Pay monthly “cost” to do merges
- You get new bugs each month

Running FreeBSD “head” lets us deliver large amounts of data to our users very efficiently, while maintaining a high velocity of feature development.

# Thank you

NETFLIX