

# Monitoring Service Architecture and Health with BPF

Jonathan Perry, Flowmill

### Agenda

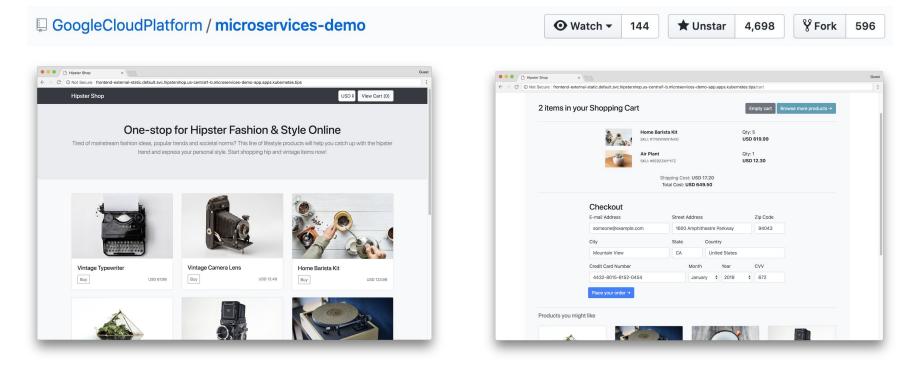
- Demo: visibility into Architecture, Health and Cost
- How that worked: Linux + Kubernetes "Flow monitoring"
- Flow vs App monitoring: Pros and Cons
- Building a complete system: **Collection & analysis architecture**
- Major challenges: Performance & completeness  $\rightarrow$  use eBPF
- Is all this really practical?: Evaluation
- Where next: Adding Application monitoring (and how)

# **Hi! I'm Jonathan Perry**

#### jperry@flowmill.com www.flowmill.com

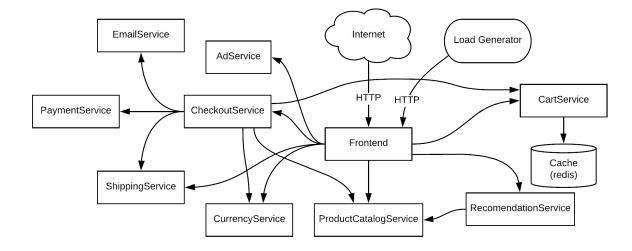
- Government: large-scale deployments
- MIT PhD: extreme monitoring systems
  - prod at Facebook
- Flowmill: CEO

# Demo application

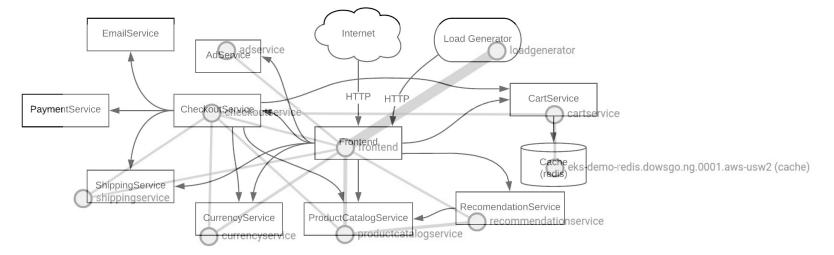


NAME	READY	STATUS	RESTARTS	AGE
adservice-6cd6965787-lmjrl	1/1	Running	0	1d
cartservice-75f55fbc45-dv85s	1/1	Running	22	1d
checkoutservice-6848667dd7-jt44p	1/1	Running	0	1d
currencyservice-668f49f985-l42s6	1/1	Running	0	1d
emailservice-796bb9588b-cfqcp	1/1	Running	0	1d
frontend-6dcd4969b4-v2vqq	1/1	Running	0	14h
loadgenerator-54d77df7b-tgnwc	1/1	Running	0	1d
paymentservice-548657568f-p5hb7	1/1	Running	0	1d
productcatalogservice-7b94dfb45c-9gz7k	1/1	Running	0	1d
recommendationservice-5fb85f46df-hwj5s	1/1	Running	0	1d
<pre>shippingservice-5f5d75bf65-v5p7f_</pre>	1/1	Running	0	1d

this shows the components, but how do they interact?  $\rightarrow$  Demo

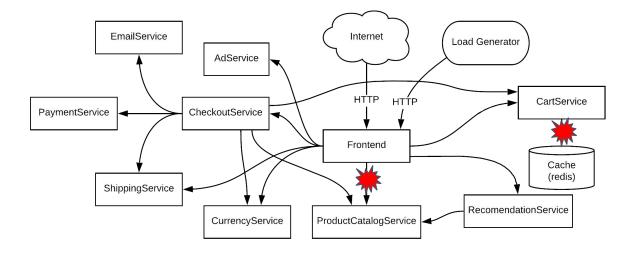






Is the mental model in touch with reality? Just deployed, are old dependencies ok? New dependency? HA: what zones are communicating?

### Visibility #2: Health



- Demo: Detecting service degradation Demo: Detecting security group misconfiguration •

### Visibility #3: Cost

Top service bandwidth consumption

- per Node
- across Zones  $\rightarrow$  Demo
- across Regions

No.         Construction         Description         Description <thdescription< th=""> <thde< th=""><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></thde<></thdescription<>											
TCP Throughput         TCP Throughput           2000         00000         000						• =					
Total         Total <th< th=""><th>•••</th><th>TCP Throughp</th><th>ut</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th><th>,</th></th<>	•••	TCP Throughp	ut								,
$I_{1} = 1 \\ I_{2} = 1 \\ I_{2$		"Дууу	Marina	www.	MMM	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	MM	www	hum	mm	W
Mark         Mark <th< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></th<>											
1/2         1/2         0	sraph	2КВ									
Image: Constraint of the		M	www.ww	mmm	mmm	mmm.	mon	mm	mm	mon	w
Normal         Op/S         <	ashooards	1KB and the		manner	man	- mana	ann	- man	amaha	man	m
Nume         0/0 <td></td> <td>m</td> <td>and the second</td> <td>A a billion and</td> <td></td> <td>min</td> <td>min</td> <td>and the</td> <td>charde</td> <td>participant of the second</td> <td>1</td>		m	and the second	A a billion and		min	min	and the	charde	participant of the second	1
Image: Section of the sectin of the section of the section			A share a share				100 C	and the		Alexander a	
Image: space	Aonitors		:00	02:00	04:00	A	06:00	A 11	08:00	10	:00
<ul> <li>publicationgrave</li> <li>westb3</li> <li>formid</li> <li>westb3</li> <li>formid</li> <li>swestb3</li> <li>formid</li> <li>swestb3</li></ul>									1	1.0	
1         - formid         user12x         pmdcdzaluger/u         user131         1.369         1.563         6.2344         1.63           Image: State St											
-         omecycycrei webili         formid         www124         507         1.182         597.39         2.9.86         507.19           Sec         omecycycrei webili         control         www124         0.000         517.11         1.182         597.39         2.9.86         557.11           -         omecycycrei webili         www124         0.000         517.11         51.40         55.11           -         omecycycrei webili         www125         omecyclearce         www126         1.000         51.51           -         omecyclearce         www126         0.000         52.36         51.61         51.61           -         omecyclearce         www126         0.000         52.36         51.61         51.61           -         omecyclearce         www126         0.000         52.36         72.37         51.84         51.84         55.64           -         omecyclearce         www126         www126         www126         52.36         52.36         52.36         52.36         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46         55.46											
2         -         formad         usere12x         ournersystemic         usere13x         110         H 2x1         42.48         34.48         54.58           2         -         formad         usere13x         chemical strategie         110.58         F2.28         42.44         114.88         114.84           -         declosateric         usere13x         chemical strategie         110.58         F2.28         42.44         114.84         64.64           -         declosateric         usere13x         chemical strategie         usere13x         126.38         77.79         42.34         14.84         45.64           -         declosateric         usere13x         chemical strategie         usere13x         126.38         77.79         42.34         14.84         45.64           -         declosateric         usere13x         chemical strategie         usere13x         127.34         14.84         15.84         45.24           -         declosateric         usere13x         chemical strategie         17.23         44.44         13.84         14.84         26.34           -         declosateric         usere13x         chemical strategie         17.23         44.44         13.84         14.94											
-         omergenerie         usest23         ordexizativic         usest23         10.05         #7.42         42.44         20.966         55.64           -         omergenerie         usest23         constraints         usest23         20.966         75.64           -         otherationativic         usest23         constraints         usest23         20.966         75.64           -         otherationativic         usest23         constraints         usest23         20.966         75.96         45.96           -         otherationativic         usest23         constraints         usest23         10.96         46.92         45.98         45.96           -         otherationativic         usest23         constraints         usest23         10.96         45.98         10.96         45.98			us-west-2a			1.3KB	1.8KB	1.5KB	62.8MB	1.6KB	
2         -         declostanterio:         useret3:         contemporterio:         useret3:         72.09         72.09         73.09         11.8.04         46.48           2         -         declostanterio:         useret3:         declostanterio:         useret3:         72.09         73.09         11.8.04         45.48           -         declostanterio:         useret3:         catativitic         useret3:         72.09         43.09         11.8.04         45.49           -         declostanterio:         useret3:         catativitic         useret3:         73.09         45.04         73.09         11.8.04         45.49           -         declostanterio:         useret3:         catativitic         useret3:         73.09         15.09         13.09         20.29           -         declostanterio:         useret3:         formation:         useret3:         13.09         20.09         13.09         20.09         13.09         13.09         13.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09         20.09	_	- currencys	us-west-2a iervice us-west-2b	frontend	us-west-2a	1.3KB 800.7B	1.8KB 1.1KB	1.5KB 959.7B	62.8MB 39.8MB	1.6KB 987.1B	
		<ul> <li>ourrencya</li> <li>frontend</li> </ul>	us-west-2a us-west-2b us-west-2a	frontend ourrencyservice	us-west-2a us-west-2b	1.3KB 800.7B 637B	1.8KB 1.1KB 1KB	1.5KB 959.7B 845.8B	62.8MB 39.8MB 34.4MB	1.6KB 987.1B 853B	
checkoutannica uswest-2a curtainvica uswest-2b 271.98 543.98 577.58 14.6MB 340.98     softwartice uswest-2b frontend uswest-2a 377.28 440.98 329.50 14.6MB 329.99     orderstandiscoutance uswest-2b frontend uswest-2a 377.28 440.98 297.99 297.98 14.6MB 329.99	<b>a</b>	<ul> <li>currencys</li> <li>frontend</li> <li>currencys</li> </ul>	us-west-2a us-west-2b us-west-2a us-west-2a	frontend ourrencyservice checkoutservice	us-west-2a us-west-2b us-west-2a	1.3KB 800.7B 637B 310.5B	1.8KB 1.1KB 1KB 874.2B	1.5KB 959.7B 845.8B 482.4B	62.8MB 39.8MB 34.4MB 20.8MB	1.6KB 987.1B 853B 516.5B	
<ul> <li>adservice us-west-2b frontend us-west-2a 317.28 404.98 338.58 14.5MB 358.96</li> <li>anotheticatelecomics us-west-2h dockstoperics us-west-2h 194.98 403.98 202.08 14.3MB 254.58</li> </ul>	8	<ul> <li>currencys</li> <li>frontend</li> <li>currencys</li> <li>checkouts</li> </ul>	us-west-2a us-west-2b us-west-2b us-west-2a vervice us-west-2a service us-west-2a	frontend ourrencyservice checkoutservice ourrencyservice	us-west-2a us-west-2b us-west-2a us-west-2b	1.3KB 800.78 6378 310.58 280.38	1.8KB 1.1KB 1KB 874.2B 787.9B	1.5KB 959.7B 845.8B 482.48 433.9B	62.8MB 39.8MB 34.4MB 20.8MB 18.8MB	1.6KB 987.1B 8538 516.58 466.88	
- productivity/operation (county) // deskusteration (county) // 19/192 5/0/20 00/201 14/048 25/158	<b>e</b>	<ul> <li>currencys</li> <li>frontend</li> <li>currencys</li> <li>checkouts</li> <li>checkouts</li> </ul>	us-west-2a us-west-2b us-west-2b us-west-2a service us-west-2a service us-west-2a	frontiend ourrencyservice checkoutservice ourrencyservice shippingservice	us-west-2a us-west-2b us-west-2a us-west-2b us-west-2b	1.3KB 800.78 6378 310.58 280.38 363.98	1.8KB 1.1KB 1KB 874.28 787.98 660.28	1.5KB 959.7B 845.8B 482.4B 433.98 475.3B	62.8MB 39.8MB 34.4MB 20.8MB 18.8MB 18.8MB 18.4MB	1.6KB 987.1B 8538 516.5B 466.88 456.68	
<ul> <li>productoatalogservice us west 2b</li> <li>checkoutservice us west 2a</li> <li>194.88</li> <li>620.28</li> <li>302.98</li> <li>14.3MB</li> <li>354.58</li> </ul>	<b>e</b> ,	<ul> <li>currencya</li> <li>frontend</li> <li>currencys</li> <li>checkouts</li> <li>checkouts</li> <li>checkouts</li> </ul>	us-west-2a wrvice us-west-2b us-west-2b wrvice us-west-2b service us-west-2a service us-west-2a service us-west-2a	frontiand ourrencyservice checkoutservice ourrencyservice shippingservice cartaervice	us-west-2a us-west-2b us-west-2a us-west-2b us-west-2b us-west-2b	1.3KB 800.7B 637B 310.5B 280.38 363.98 271.08	1.8KB 1.1KB 1KB 874.2B 787.98 660.28 543.98	1.5KB 959.7B 845.8B 482.4B 433.9B 475.3B 377.5B	62.8MB 39.8MB 34.4MB 20.8MB 18.8MB 18.4MB 14.6MB	1.6KB 987.1B 853B 516.5B 466.8B 456.6B 362.9B	
AGOUR	<b>e</b> ;	<ul> <li>currencys</li> <li>frontend</li> <li>currencys</li> <li>checkouts</li> <li>checkouts</li> <li>checkouts</li> <li>adservice</li> </ul>	us-west-2a us-west-2a us-west-2a vervice us-west-2a service us-west-2a service us-west-2a service us-west-2a	frontend ourrencyservice checkoutservice ourrencyservice shippingservice carteirvice frontend	us-west-2a us-west-2b us-west-2a us-west-2b us-west-2b us-west-2b us-west-2a	1.3KB 800.78 6378 310.58 280.38 363.98 271.08 317.28	1.8KB 1.1KB 1KB 874.2B 787.98 660.28 543.98 404.98	1.5KB 959.7B 845.8B 482.4B 433.9B 475.3B 377.5B 338.5B	62.8MB 39.8MB 34.4MB 20.8MB 18.8MB 18.4MB 14.6MB 14.5MB	1.6KB 987.1B 853B 516.5B 466.3B 456.6B 362.9B 358.9B	

								6	
	From Service	From Zone	To Service	To Zone	Min	Max	Current	↓ Total	Average
-	productcatalogservice	us-west-2b	frontend	us-west-2a	2.8KB	3.7KB	3.2KB	136MB	3.4KB
-	frontend	us-west-2a	productcatalogservic	us-west-2b	1.3KB	1.8KB	1.5KB	62.8MB	1.6KB
-	currencyservice	us-west-2b	frontend	us-west-2a	800.7B	1.1KB	959.7B	39.8MB	987.1B
-	frontend	us-west-2a	currencyservice	us-west-2b	637B	1KB	845.8B	34.4MB	853B
-	currencyservice	us-west-2b	checkoutservice	us-west-2a	310.5B	874.2B	482.4B	20.8MB	516.5B
-	checkoutservice	us-west-2a	currencyservice	us-west-2b	280.3B	787.9B	433.9B	18.8MB	466.8B
-	checkoutservice	us-west-2a	shippingservice	us-west-2b	363.9B	660.2B	475.3B	18.4MB	456.6B
-	checkoutservice	us-west-2a	cartservice	us-west-2b	271.9B	543.9B	377.5B	14.6MB	362.9B
-	adservice	us-west-2b	frontend	us-west-2a	317.2B	404.9B	338.5B	14.5MB	358.9B
-	productcatalogservice	us-west-2b	checkoutservice	us-west-2a	194.8B	620.2B	302.9B	14.3MB	354.5B
-	frontand	ue weet ?e	oortoonvice	us west 2h	000 1D	240 4D	ac coc	10 0M/D	00 700

#### How: Flow data

Linux:	Timestamp 1418530010	Source	Destination         Ports         Bytes         Drops         RTT           172.31.16.21         20641         22         4249         2         4 ms	
K8s:	IP	Pod	Image Tag Zone	
	172.31.16.139 172.31.16.21	frontend checkoutservi	frontend-image v1.16 us-west-1c ice checkout-image v2.12a us-west-1a	
Joined:	Timestamp	Source	Destination   Ports   Bytes   Drops   RTT	
	1418530010	frontend frontend-image v1.16 us-west-1c	checkout 20641 22 4249 2 4 ms checkout-image v2.12a us-west-1a	

#### Getting Flow Data \$ kubectl describe pod \$POD Name: А staging Namespace: . . . Running Status: 100.101.198.137 IP: Controlled By: ReplicaSet/A A→X # PID=`docker inspect -f '{{.State.Pid}}' \$CONTAINER` \ A.X nsenter -t SPID -n ss -ti FSTAB 100.101.198.137:34940 100.65.61.118:8000 Ø Ø iptables A.B cubic wscale:9.9 rto:204 rtt:0.003/0 mss:1448 cwnd:19 ssthresh:19 bytes\_acked:2525112 segs\_out:15664 segs\_in:15578 data\_segs\_oui.15662\_send 73365.5Mbps lastsnd:384 lastrcv:10265960 lastack:384 rcv\_space:29200 minrtt:0.002 A→B # conntrack -L <u>6 86399 ESTABLISHED src=100.101.198.137 A</u> tcp dst=100.65.61.118 sport=34940 dport=8000 pod src=100.101.198.147 dst=100.101.198.137 sport=8000 dport=34940 [ASSURED] mark=0 use=1

### Flow monitoring: Pros and cons

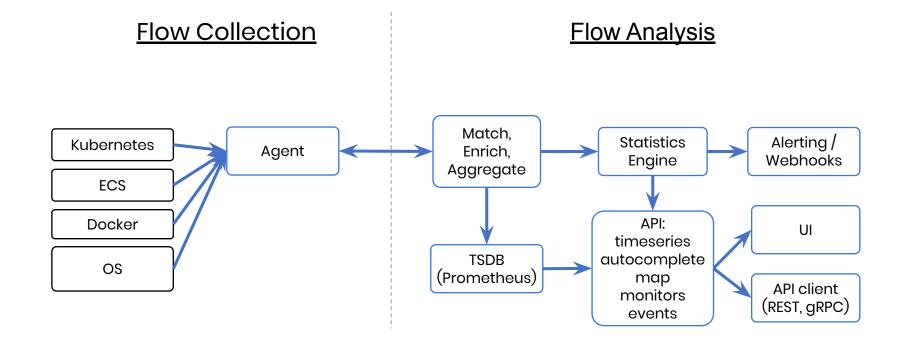
Pros:

- No code changes only use info from Linux+k8s
- 100% coverage same reason
- Small overhead few, optimizable collection points
  - more on this in "Evaluation" section
- External visibility observe managed services, APIs

Cons:

- No application-level error codes
  - only see proxies (bandwidth, rtt, drops)
  - solvable more towards end of talk

#### Flow monitoring: system architecture



#### Addressing performance & completeness

• Performance: (1) iterates all sockets, (2) built for occasional use

• Completeness: Linux CLI tools are polling based



 $\rightarrow$  Misses events between polls

#### Enter eBPF

- Linux bpf() system call since 3.18
- Run code on kernel events
- Only changes, more data
- Safe: In-kernel verifier, read-only
- Fast: JIT-compiled



Unofficial BPF mascot by Deirdré Straughan

#### $\rightarrow$ 100% coverage + no app changes + low overhead ftw!

# Using eBPF

iovisor / bcc	🛈 Watch 🗸	439	★ Star	6,735	<b>∛</b> Fork	1,130

#### Demo:

#### to run a bcc container:

```
docker run -it --rm \
    --privileged \
    -v /lib/modules:/lib/modules:ro \
    -v /usr/src:/usr/src:ro \
    -v /etc/localtime:/etc/localtime:ro \
    --workdir /usr/share/bcc/tools \
    --pid=host \
    zlim/bcc
```

https://github.com/iovisor/bcc/blob/master/QUICKSTART.md + host pid namespace

#### tcptop:

 instruments tcp\_sendmsg and tcp\_cleanup\_rbuf

```
need to be careful of races:
    # IPv4: build dict of all seen keys
    ipv4_throughput = defaultdict(lambda: [0, 0])
    for k, v in ipv4_send_bytes.items():
        key = get_ipv4_session_key(k)
        ipv4_throughput[key][0] = v.value
        ipv4_send_bytes_clear()
```

as for loop is running, kernel continues with updates, clear() throws those out.

### Evaluation: CPU overhead

using *perf* and FlameGraph[1]

- To record: perf record -a -g -e cycles -c 5000000 -- sleep 60
- **Post-process:** perf script | FlameGraph/stackcollapse-perf.pl > raw.txt
- Analyze:grep -E '(cleanup\_module|flowmill\_agent)' raw.txt | FlameGraph/flamegraph.pl > flame.svg
- → observed 0.1% 0.25% CPU overhead across deployments

Most aggressive customer load test:

	Node	Application	TCP stack	Collector
M cycles (%)	480,000 (100%)	220,775 (46%)	27135 (5.6%)	4,120 (0.86%)

[1] github.com/brendangregg/FlameGraph

#### Evaluation: Network overhead

Flow observability → monitor the flow-telemetry flows Megabytes / second

	App throughput	Flow telemetry	%
Cluster 1	186.2	0.85	0.46%
Cluster 2	217.1	2.49	1.15%
Cluster 3	249.6	0.25	0.10%
Cluster 4 (batch)	522.0	0.16	0.031%
Cluster 5	183.0	0.02	0.013%

 $\rightarrow$  Usually < 0.5% network overhead, outliers ~1%

### Evaluation: Backend QPS

Agent event counts (per second):

	TCP	UDP	NAT	process	container	DNS	Total events/s per agent
Company A	1429.2	82.0	20.8	146.5	0.014	10.5	1689.014
Company B	4017.3	89.0	-	1562.1	-	1.98	5670.38
Company C (batch)	51.0	28.8	1.05	43.8	0.55	0.5	125.7

- → For a 50-node cluster, need to process 84.4k-283.5k QPS (~20x less for batch workloads)
- → C++ analysis pipeline: hundreds of nodes w/2 second latency (thousands soon)

### Addressing the cons: application metrics

#### • eBPF supports user probes

#### $\rightarrow$ Demo

l I	nm server   grep 'net/http\.'
t	net/http.Error
t	net/http.Get
t	net/http.HandleFunc
t	net/http.Handler.ServeHTTP-fm
t	net/http.HandlerFunc.ServeHTTP
t	net/http.Header.Add
t	net/http.Header.Del
t	net/http.Header.Get
t	net/http.Header.Set
t	net/http.Header.Write
t	net/http.Header.WriteSubset
t	net/http.Header.clone
	t

<pre>\$ ./funccount -p 24503 -r './server:net/</pre>	'http\*Writer'
FUNC	COUNT
<pre>net/http.(*chunkWriter).Write</pre>	3
<pre>net/http.(*chunkWriter).close</pre>	3
net/http.(*chunkWriter).writeHeader	3
net/http.checkConnErrorWriter.Write	3
<pre>net/http.(*chunkWriter).writeHeader.func</pre>	:1 3
net/http.newBufioWriterSize	6
net/http.putBufioWriter	6
Detaching	

# Flow monitoring

#### Visibility into Architecture, Health, and Cost

- No code changes
- Negligible overhead
- Visibility into external dependencies
- Want application metrics (in progress)

Questions? (and please reach out)

Jonathan Perry <jperry@flowmill.com www.flowmill.com with eBPF