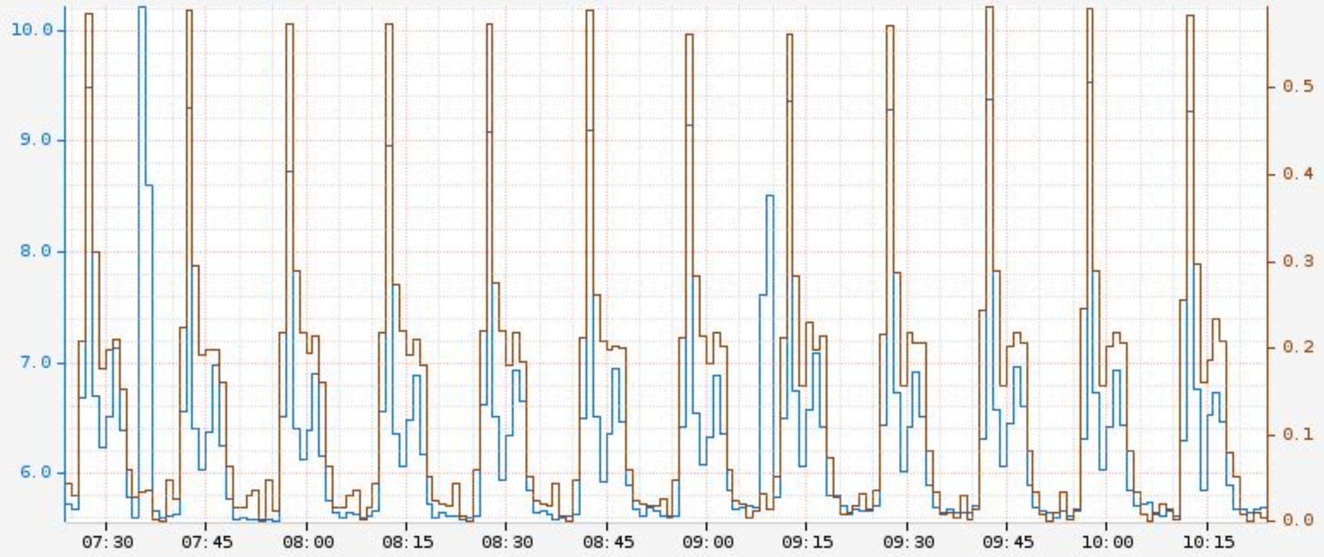
FlameCommander

Netflix's cloud profiler.

MARTIN SPIER
PERFORMANCE ENGINEER

NETFLIX

@spiermar



Axis 0

🟦 RequestStats-all-requests-_TotalTimeMillis [redacted]

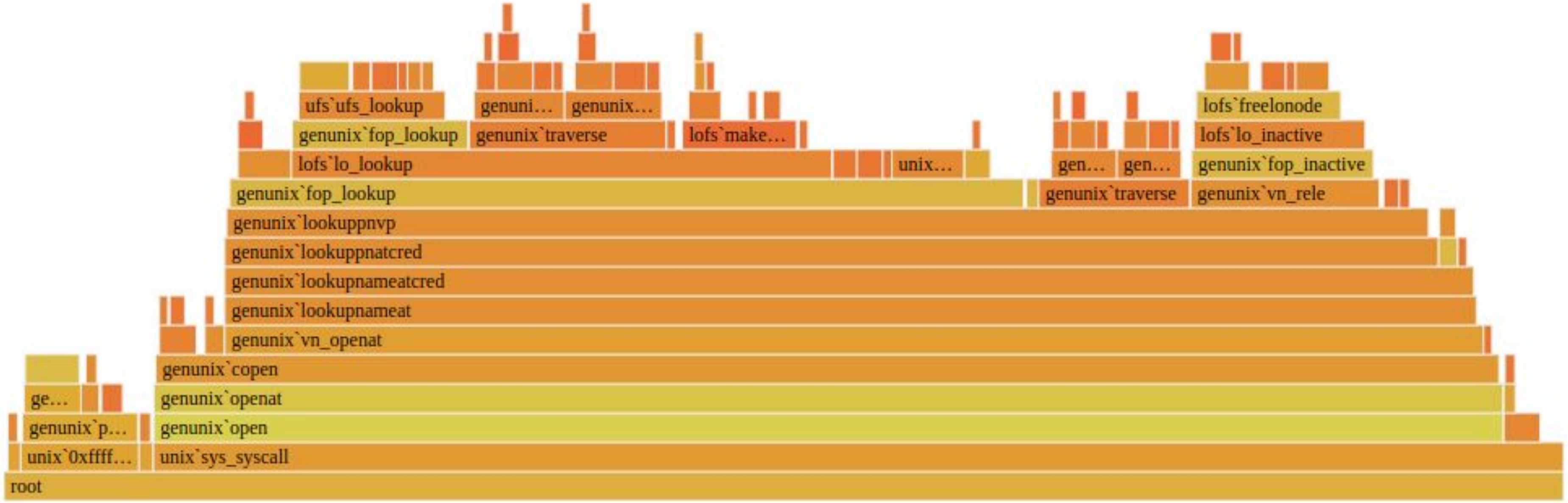
```
Max :  [redacted]   Min  :  [redacted]
Avg :  [redacted]   Last :  [redacted]
Tot :  [redacted]   Cnt  :  [redacted]
```

Axis 1

🟫 [redacted]

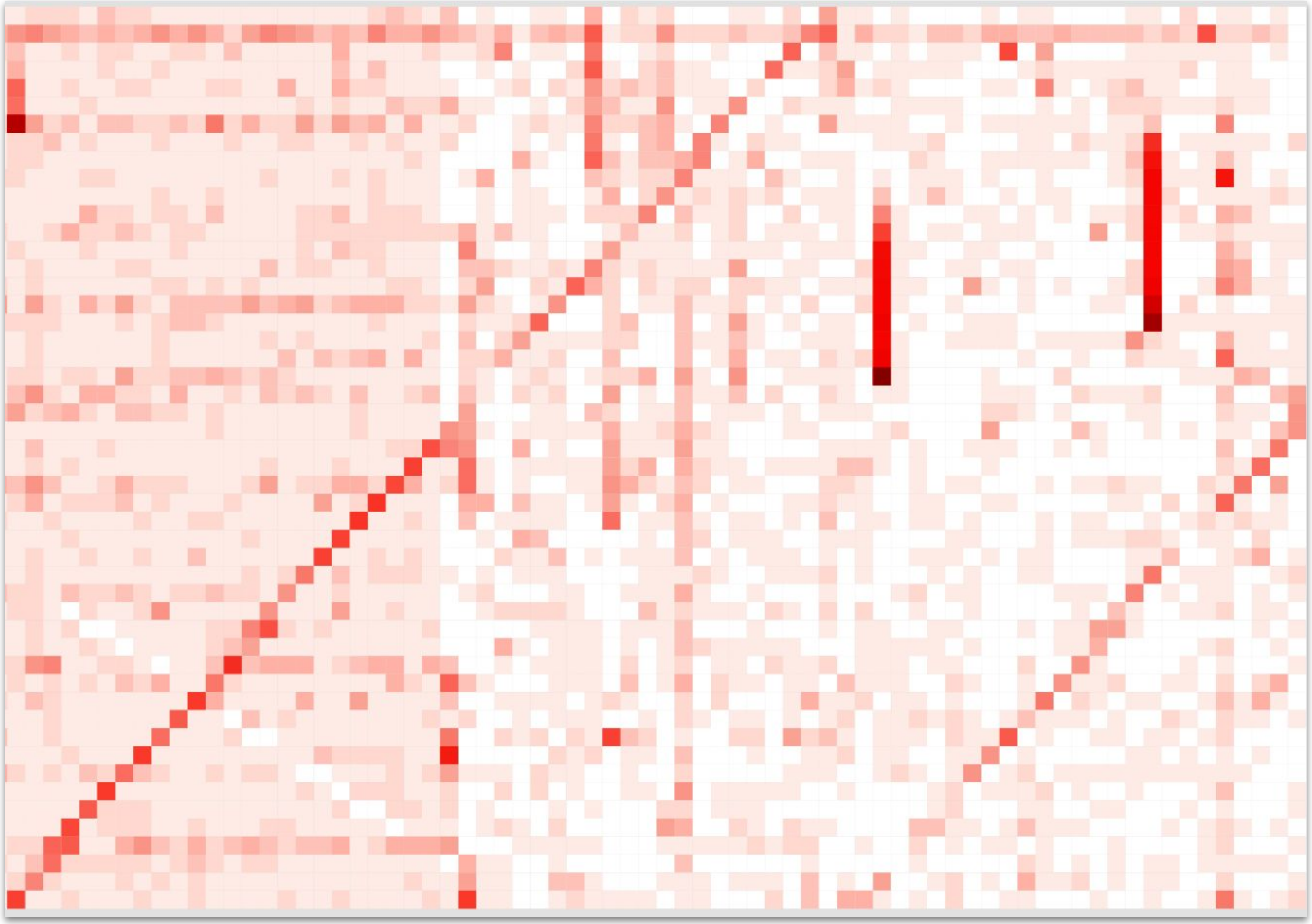
```
Max :  [redacted]   Min  :  [redacted]
Avg :  [redacted]   Last :  [redacted]
Tot :  [redacted]   Cnt  :  [redacted]
```

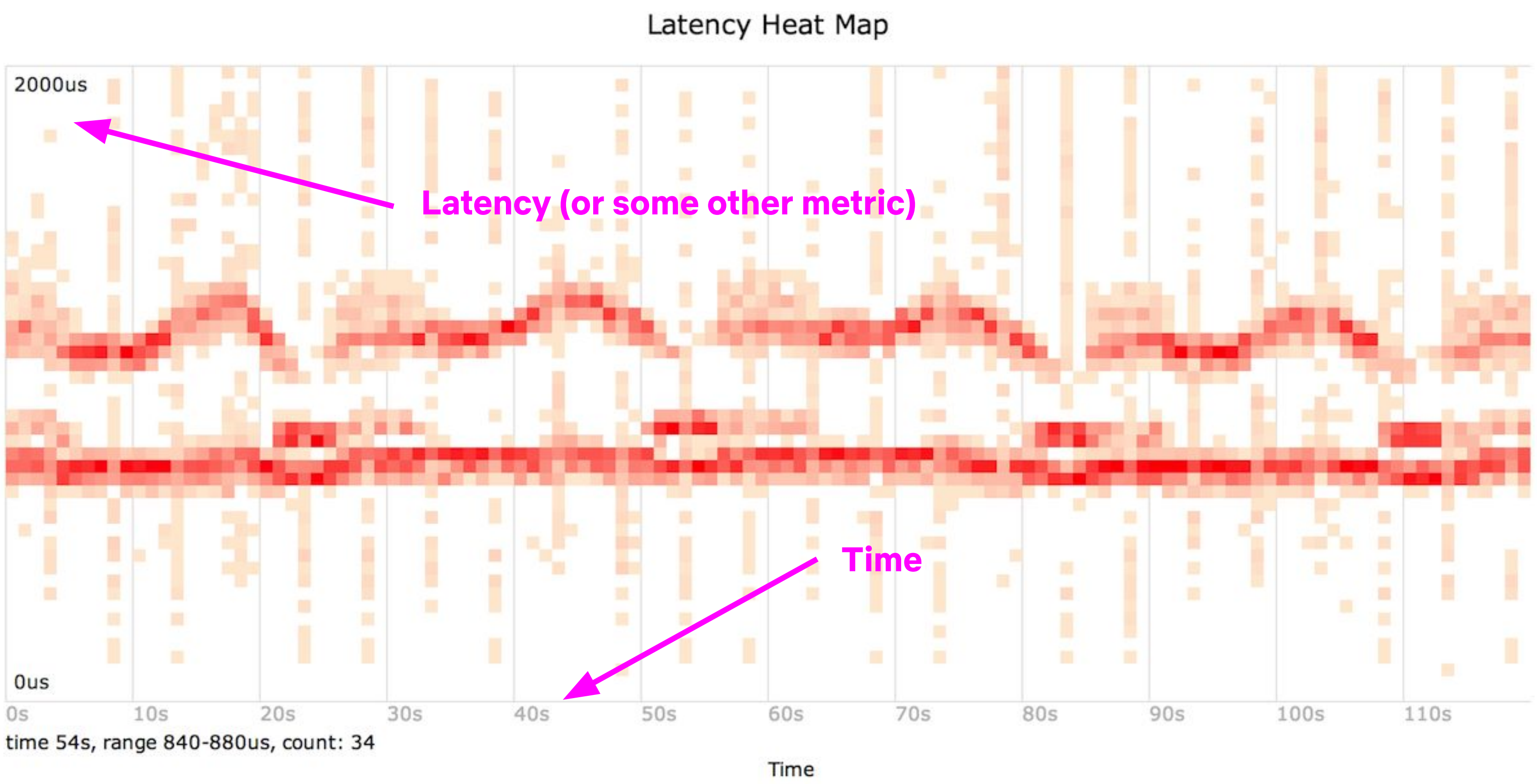
```
Frame: [redacted]
Fetch:
```

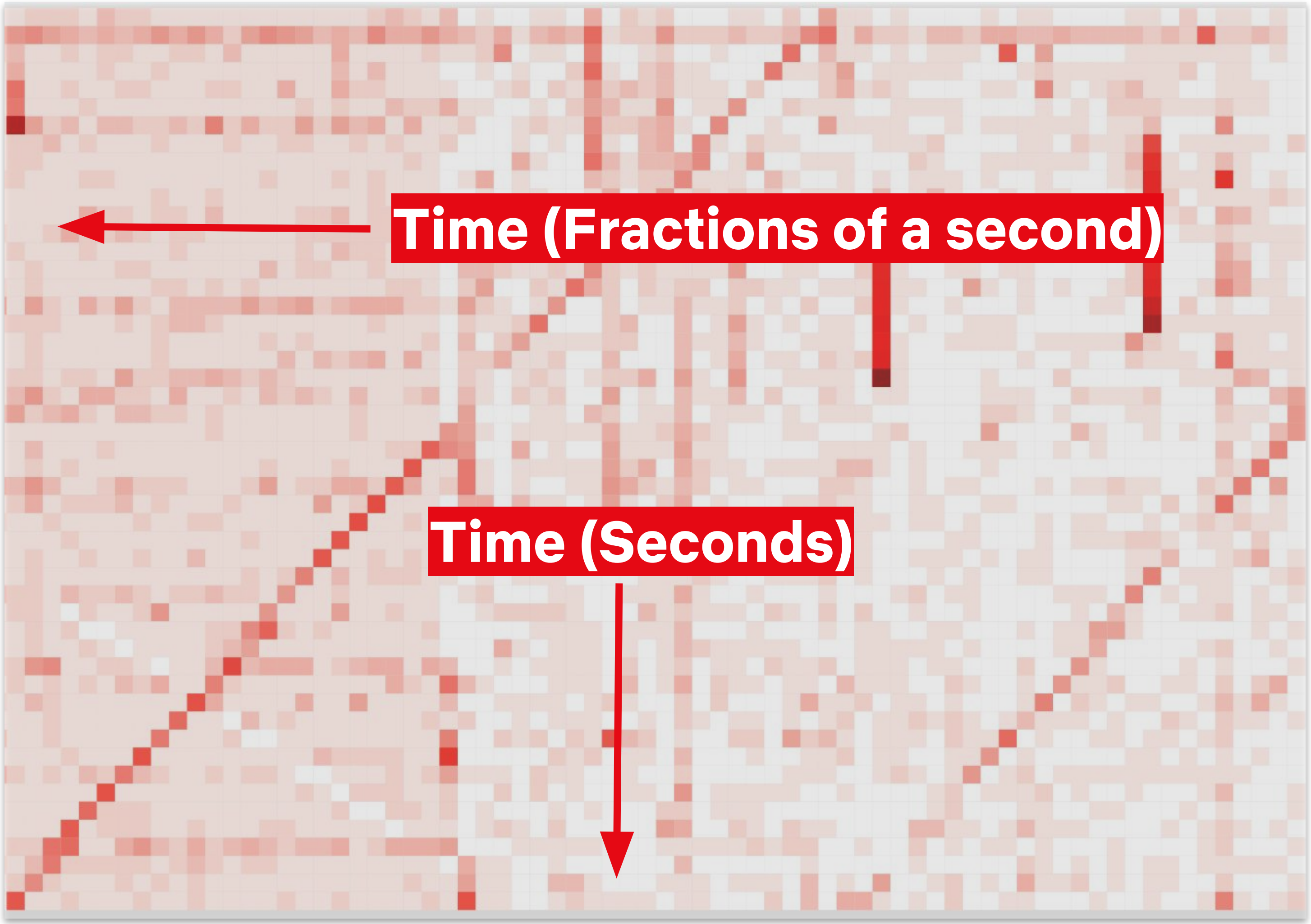

ufs`ufs_lookup
genuni…
genunix…
genunix`fop_lookup
genunix`traverse
lofs`make…
lofs`freelonode
lofs`lo_lookup
unix…
gen…
gen…
lofs`lo_inactive
genunix`fop_inactive
genunix`fop_lookup
genunix`traverse
genunix`vn_rele
genunix`lookuppnvp
genunix`lookuppnatcred
genunix`lookupnameatcred
genunix`lookupnameat
genunix`vn_openat
genunix`copen
ge…
genunix`openat
genunix`p…
genunix`open
unix`0xffff…
unix`sys_syscall
root

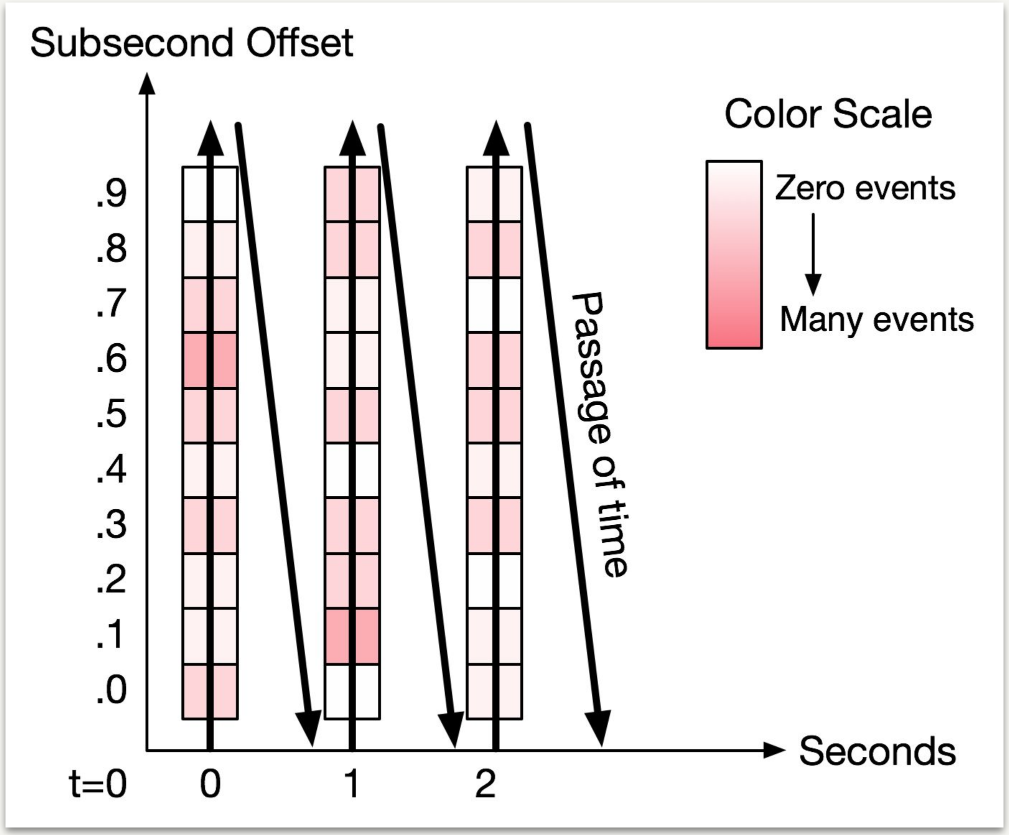
We could not “catch” the issue with a regular profile.

Flame graphs don’t have a **time dimension**, so we created a **secondary** visualization.

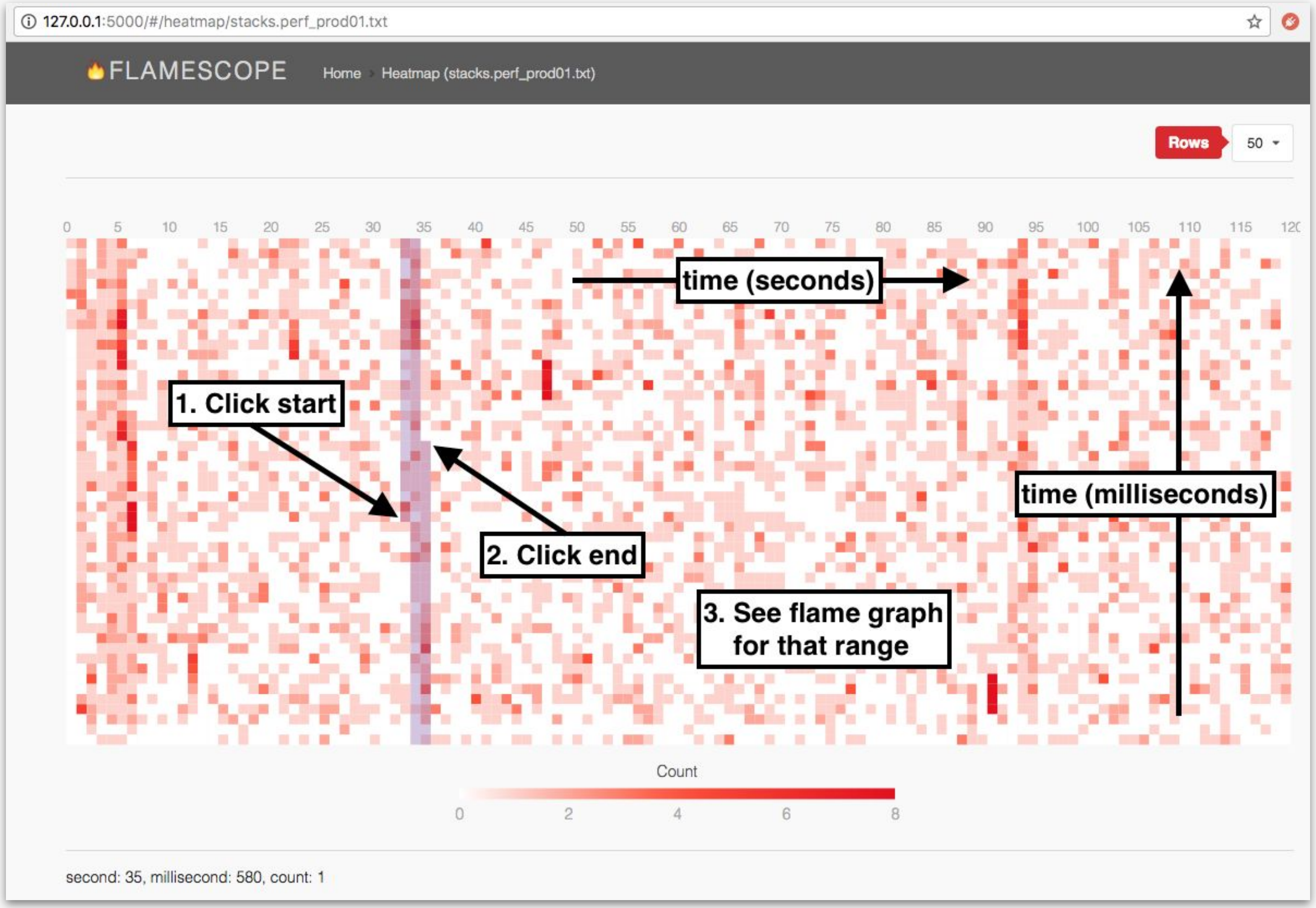



Latency Heat Map
2000us
0us
Latency (or some other metric)
Time
0s
10s
20s
30s
40s
50s
60s
70s
80s
90s
100s
110s
time 54s, range 840-880us, count: 34
Time

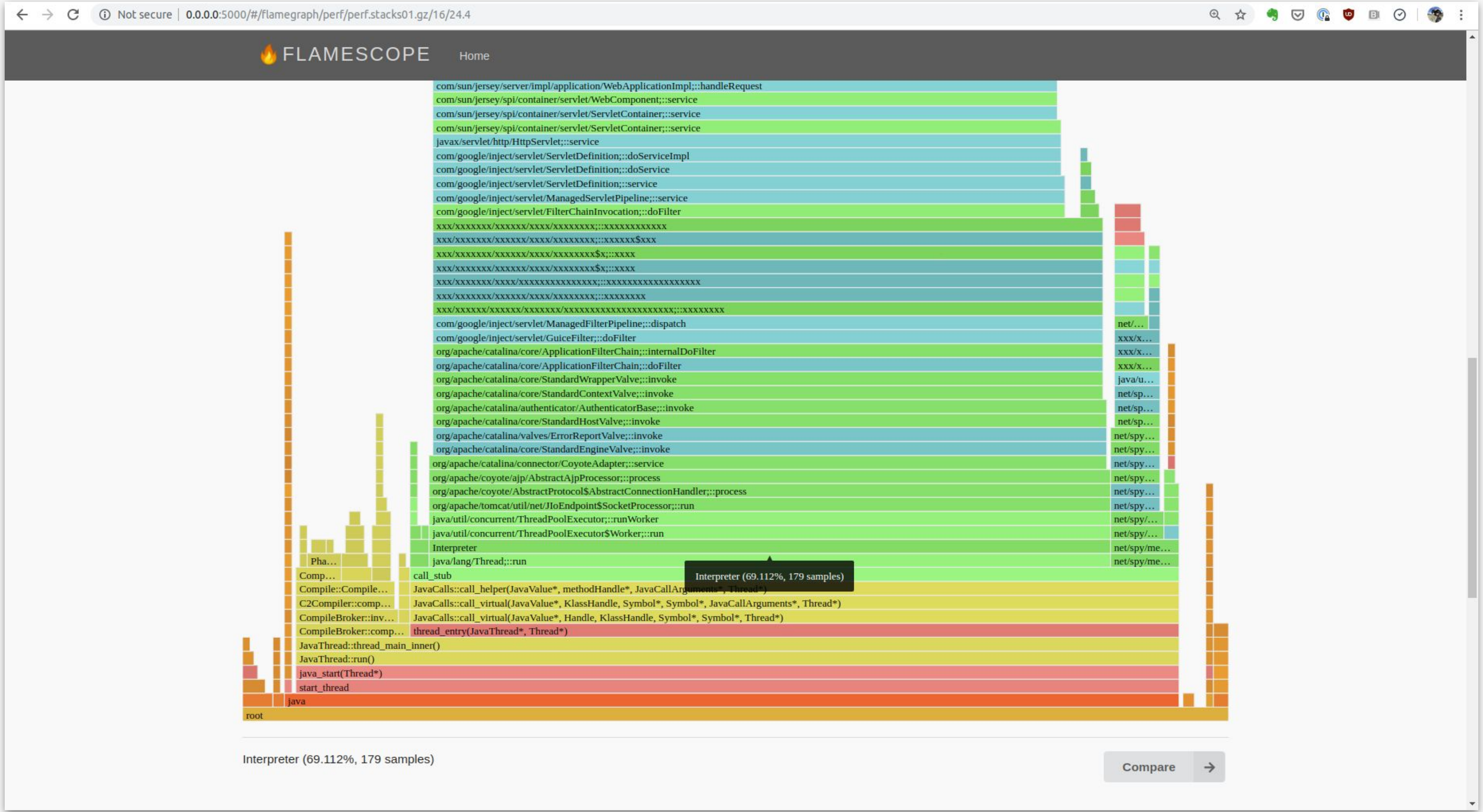

Time (Fractions of a second)
Time (Seconds)



Brendan Gregg, 2018

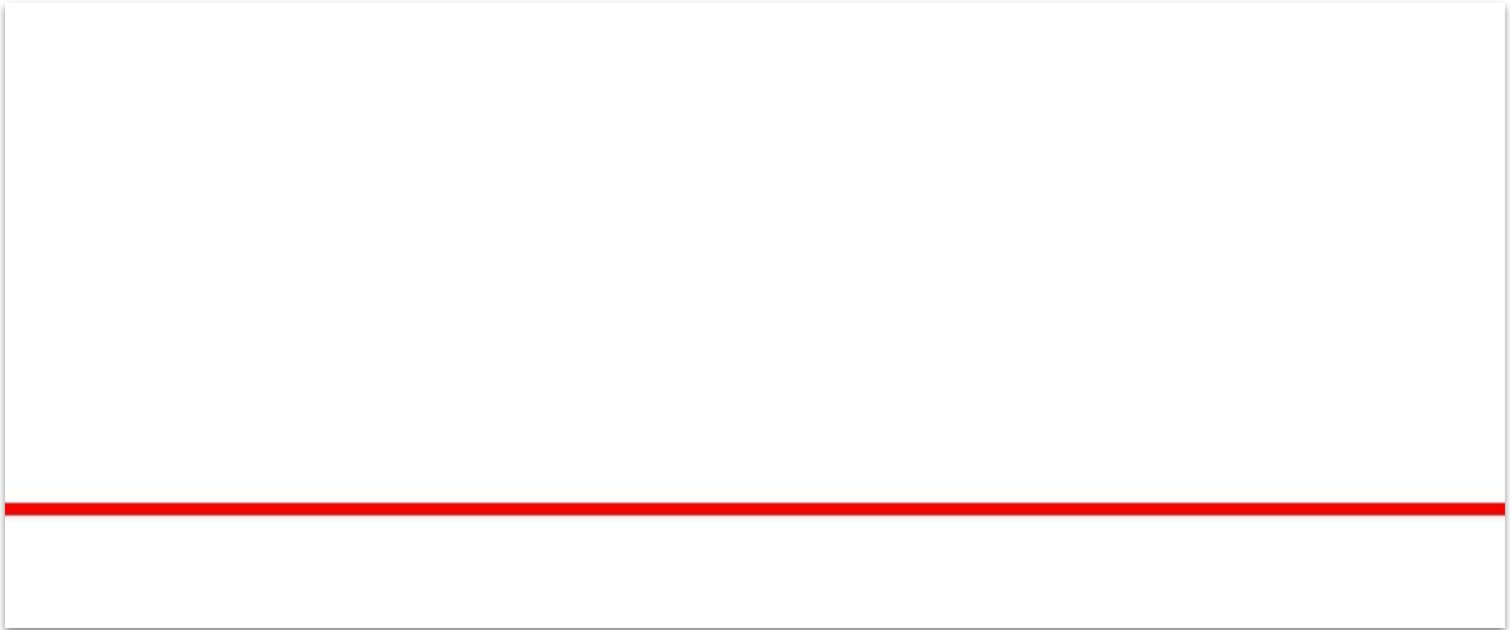


second: 35, millisecond: 580, count: 1

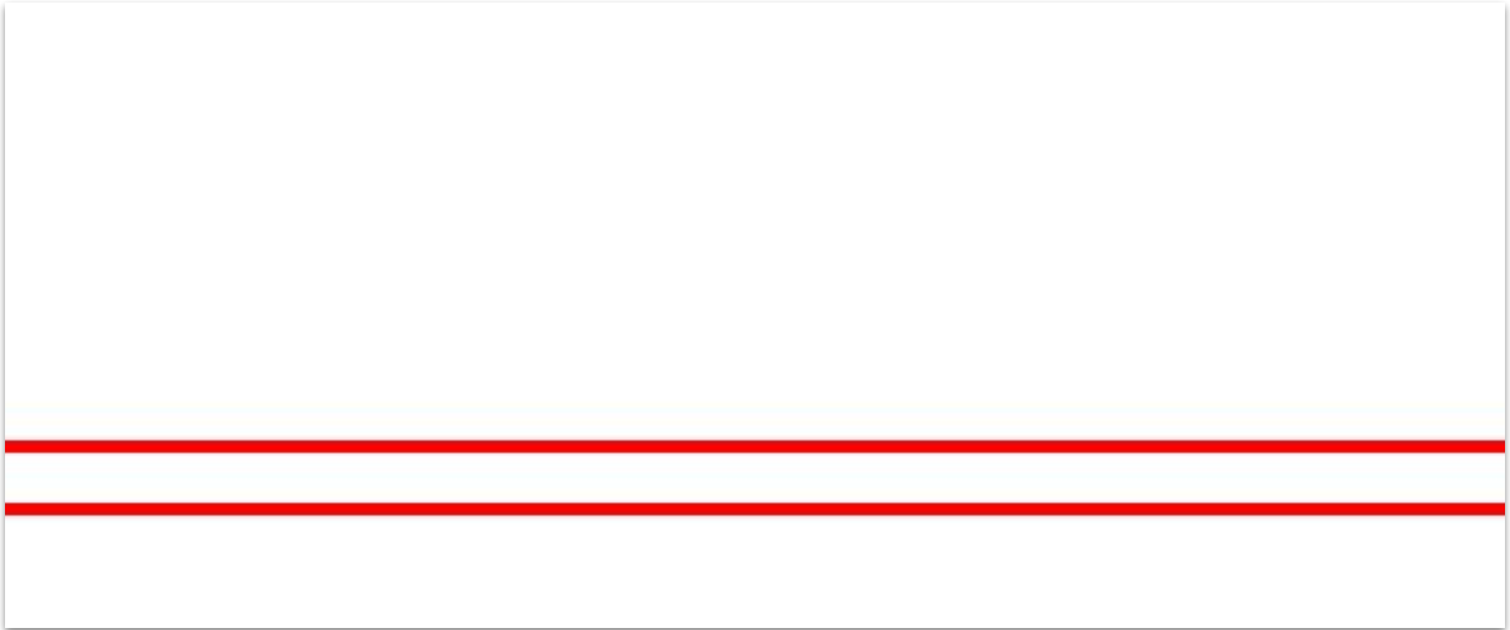

Not secure | 0.0.0.0:5000/#/flamegraph/perf/perf.stacks01.gz/16/24.4
FLAMESCOPE
Home
com/sun/jersey/server/impl/application/WebApplicationImpl;::handleRequest
com/sun/jersey/spi/container/servlet/WebComponent;::service
com/sun/jersey/spi/container/servlet/ServletContainer;::service
com/sun/jersey/spi/container/servlet/ServletContainer;::service
javax/servlet/http/HttpServlet;::service
com/google/inject/servlet/ServletDefinition;::doServiceImpl
com/google/inject/servlet/ServletDefinition;::doService
com/google/inject/servlet/ServletDefinition;::service
com/google/inject/servlet/ManagedServletPipeline;::service
com/google/inject/servlet/FilterChainInvocation;::doFilter
xxx/xxxxxxx/xxxxxx/xxxx/xxxxxxxx;::xxxxxxxxxxxx
xxx/xxxxxxx/xxxxxx/xxxx/xxxxxxxx;::xxxxxx$xxx
xxx/xxxxxxx/xxxxxx/xxxx/xxxxxxxx$x;::xxxx
xxx/xxxxxxx/xxxxxx/xxxx/xxxxxxxx$x;::xxxx
xxx/xxxxxxx/xxxx/xxxxxxxxxxxxxxx;::xxxxxxxxxxxxxxxxx
xxx/xxxxxxx/xxxxxx/xxxx/xxxxxxxx;::xxxxxxxx
xxx/xxxxxx/xxxxxx/xxxxxxx/xxxxxxxxxxxxxxxxxxxx;::xxxxxxxx
com/google/inject/servlet/ManagedFilterPipeline;::dispatch
com/google/inject/servlet/GuiceFilter;::doFilter
org/apache/catalina/core/ApplicationFilterChain;::internalDoFilter
org/apache/catalina/core/ApplicationFilterChain;::doFilter
org/apache/catalina/core/StandardWrapperValve;::invoke
org/apache/catalina/core/StandardContextValve;::invoke
org/apache/catalina/authenticator/AuthenticatorBase;::invoke
org/apache/catalina/core/StandardHostValve;::invoke
org/apache/catalina/valves/ErrorReportValve;::invoke
org/apache/catalina/core/StandardEngineValve;::invoke
org/apache/catalina/connector/CoyoteAdapter;::service
org/apache/coyote/ajp/AbstractAjpProcessor;::process
org/apache/coyote/AbstractProtocol$AbstractConnectionHandler;::process
org/apache/tomcat/util/net/JIoEndpoint$SocketProcessor;::run
java/util/concurrent/ThreadPoolExecutor;::runWorker
java/util/concurrent/ThreadPoolExecutor$Worker;::run
Interpreter
java/lang/Thread;::run
call_stub
JavaCalls::call_helper(JavaValue*, methodHandle*, JavaCallArguments*, Thread*)
JavaCalls::call_virtual(JavaValue*, KlassHandle, Symbol*, Symbol*, JavaCallArguments*, Thread*)
JavaCalls::call_virtual(JavaValue*, Handle, KlassHandle, Symbol*, Symbol*, Thread*)
thread_entry(JavaThread*, Thread*)
net/...
xxx/x...
java/u...
net/sp...
net/spy...
net/spy/...
net/spy/me...
Pha...
Comp...
Compile::Compile...
C2Compiler::comp...
CompileBroker::inv...
CompileBroker::comp...
JavaThread::thread_main_inner()
JavaThread::run()
java_start(Thread*)
start_thread
java
root
Interpreter (69.112%, 179 samples)
Interpreter (69.112%, 179 samples)
Compare

The new visualization helped us solve the intermittent behavior issue (and a few others).

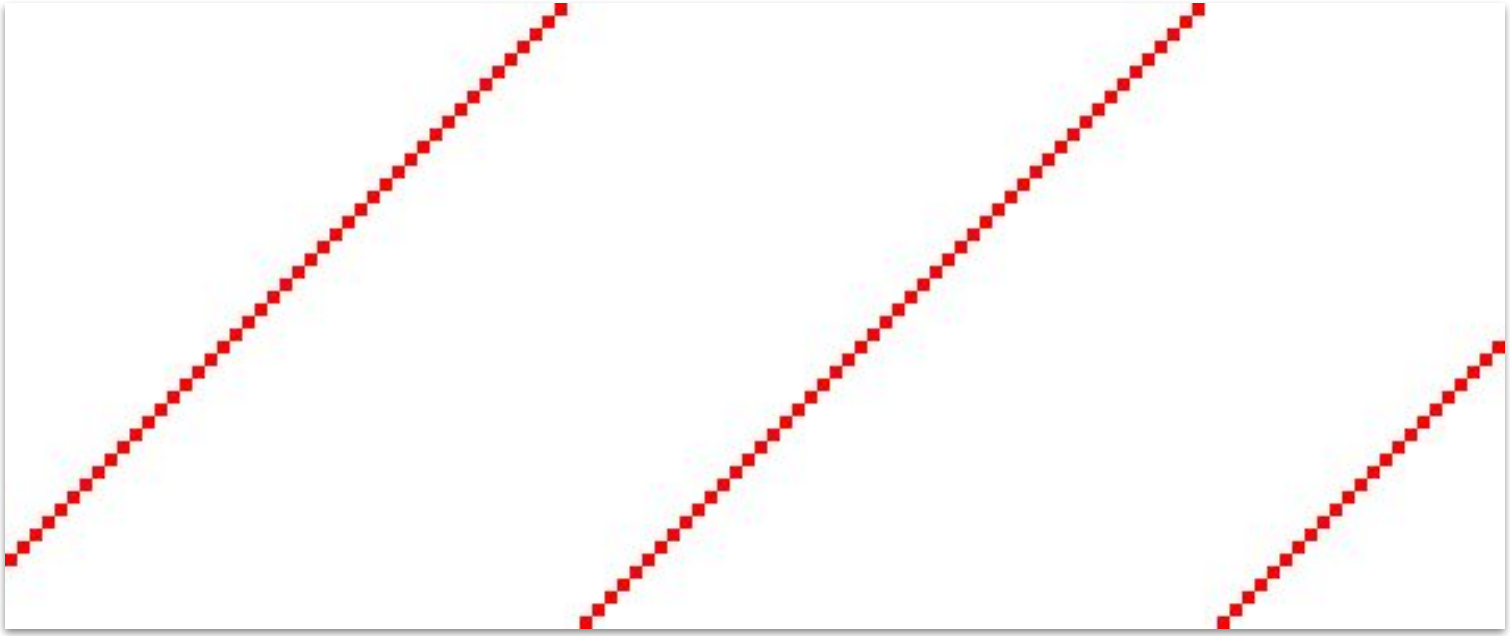
Plotting the profile as a **heatmap** also enabled us to easily identify **patterns** in them.



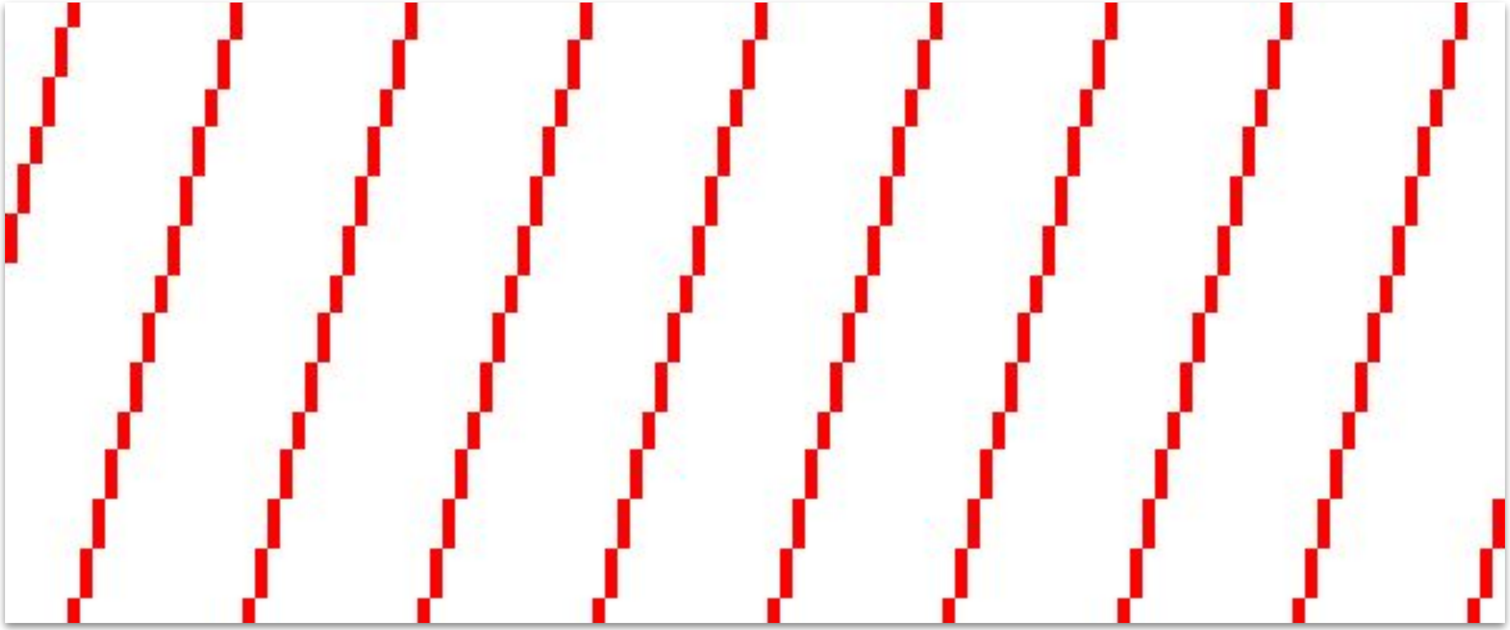
Single thread, once a sec.



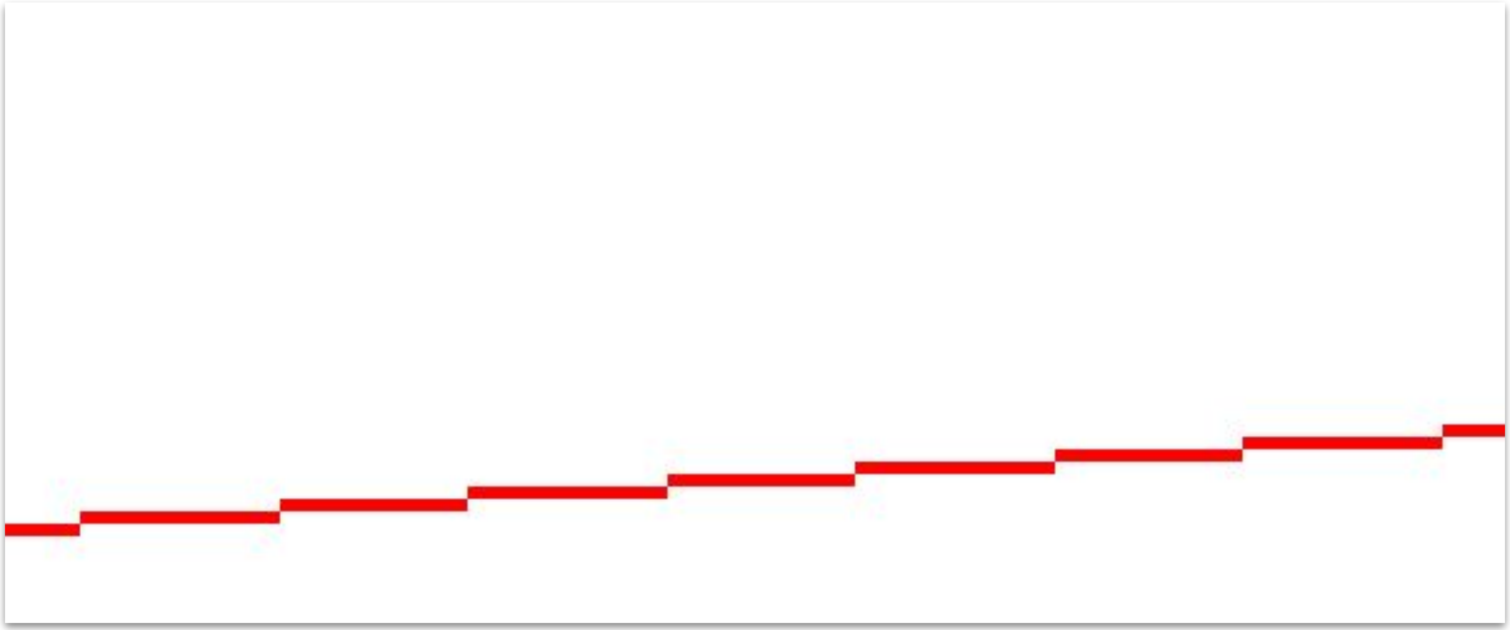
Two threads.



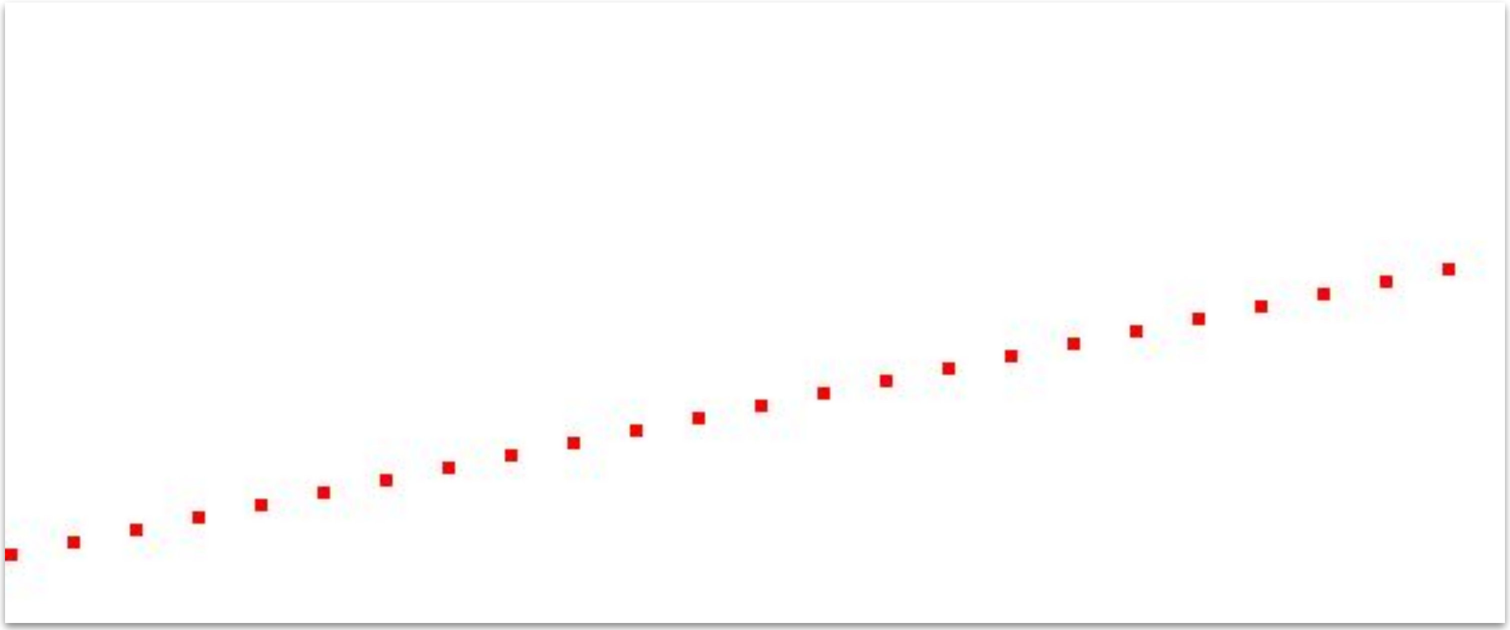
One busy-wait thread, once a sec.



One heavy busy-wait thread.

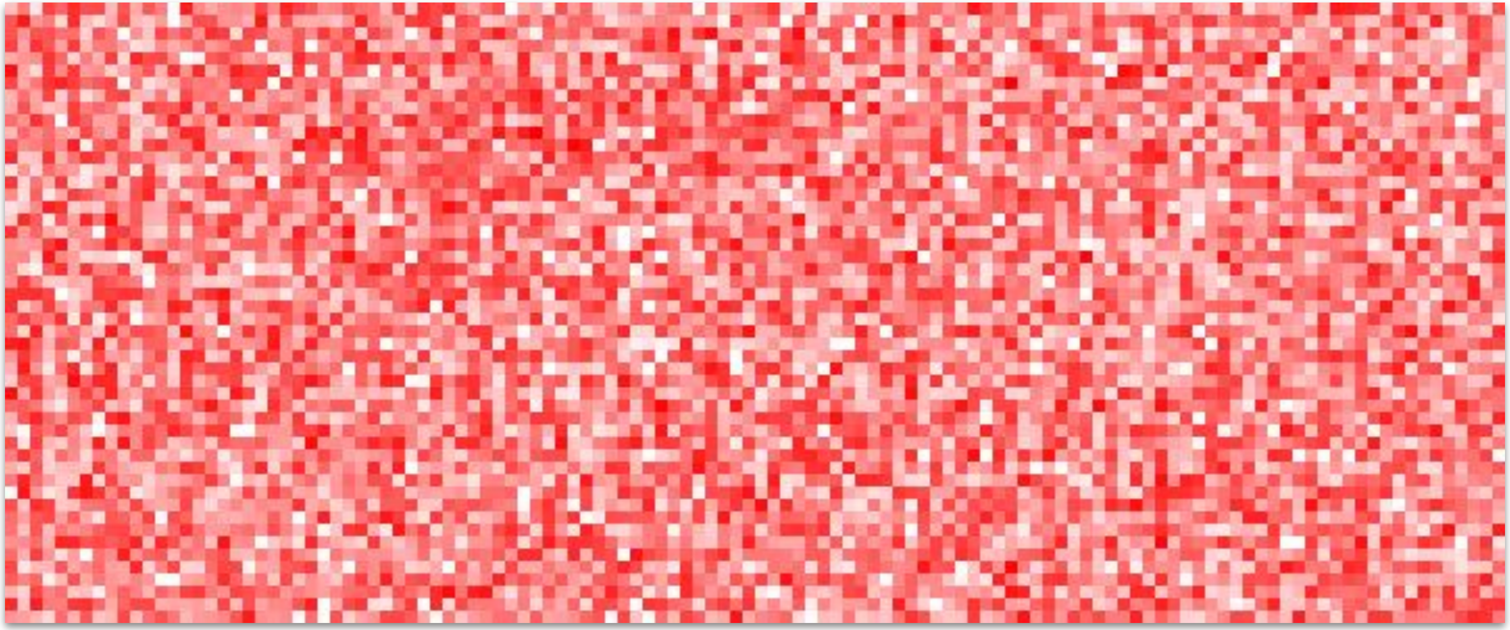


One busy-wait thread, doing less.

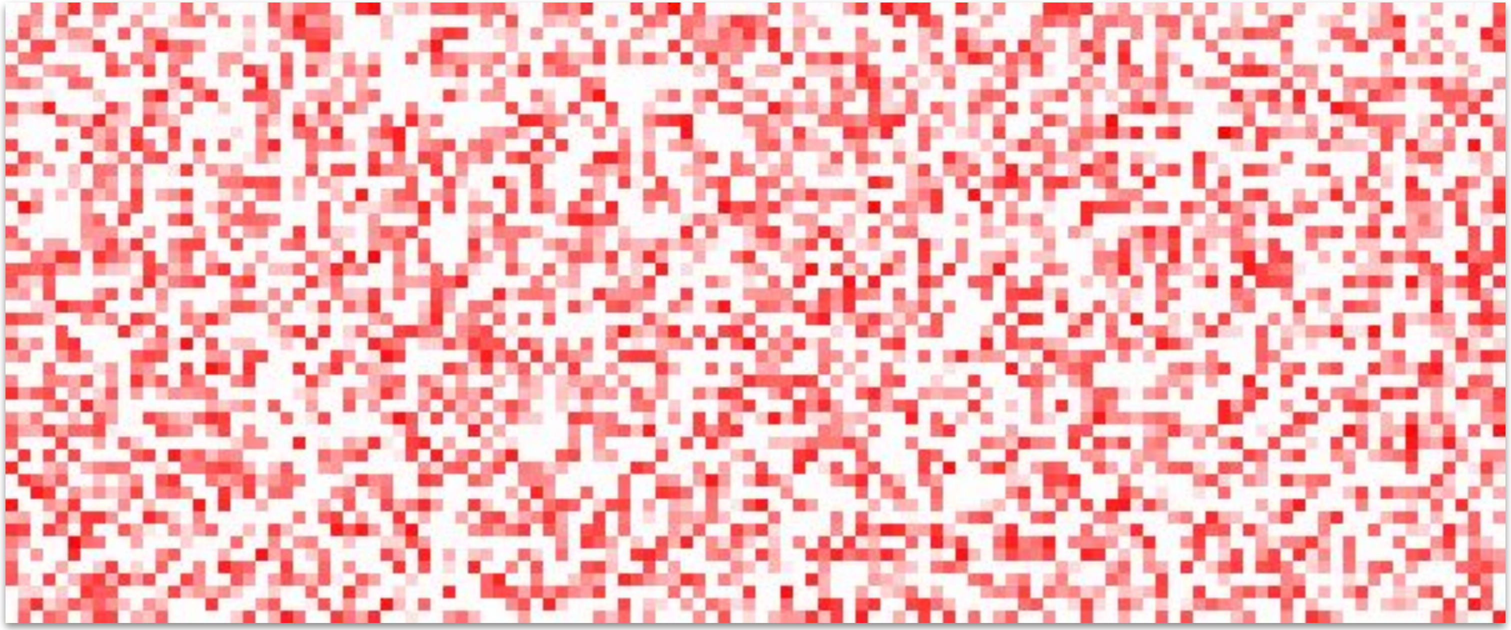


One busy-wait thread, every 5s.

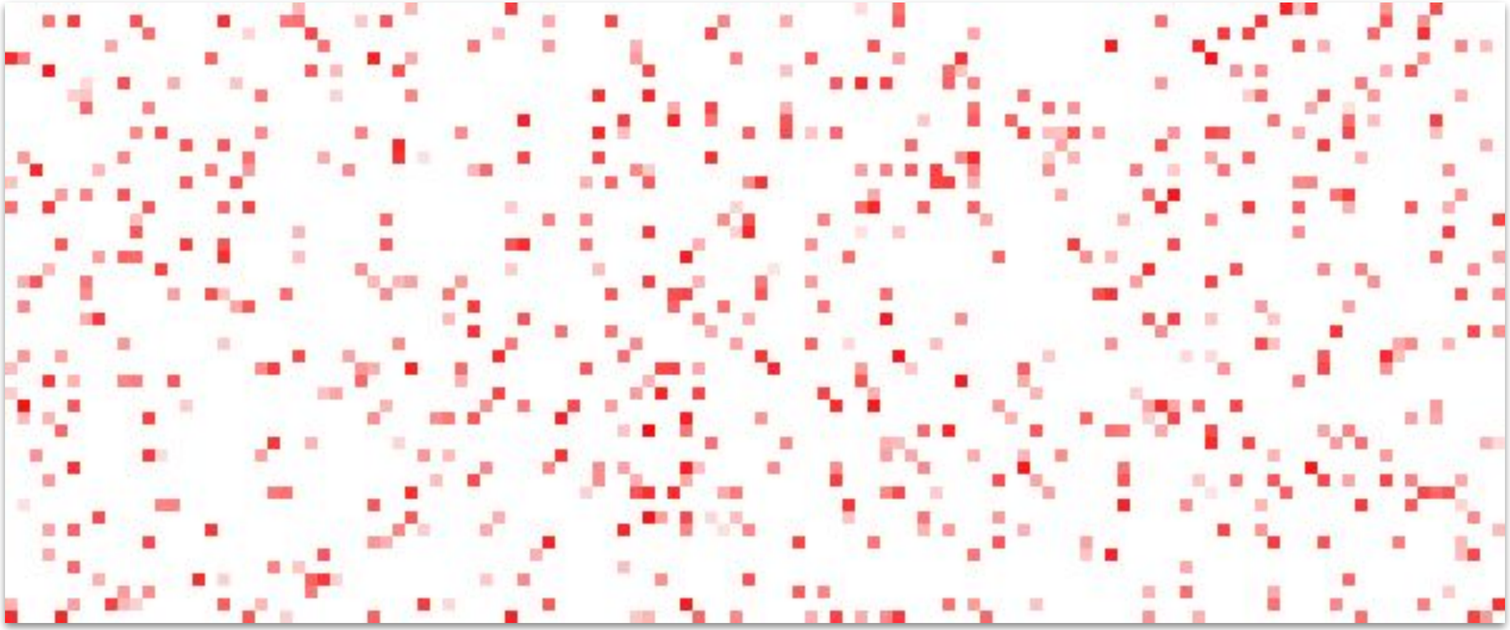
100% CPU.



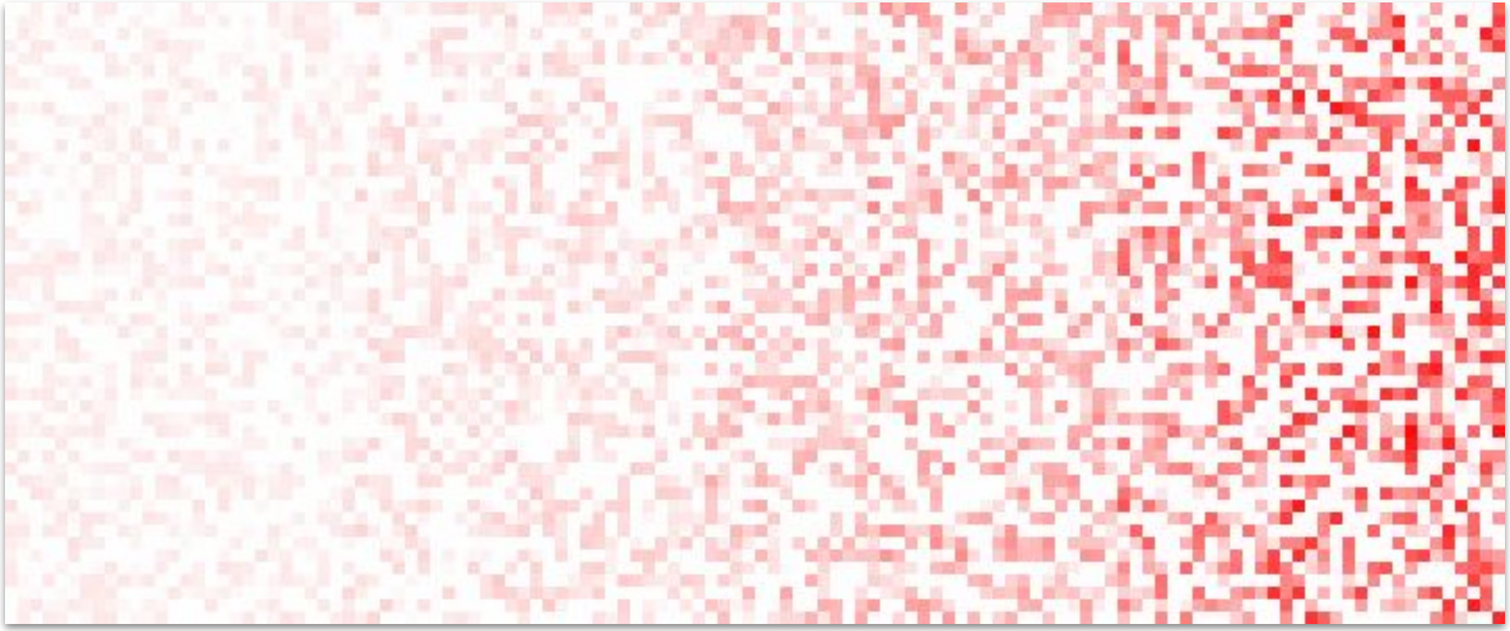
50%.



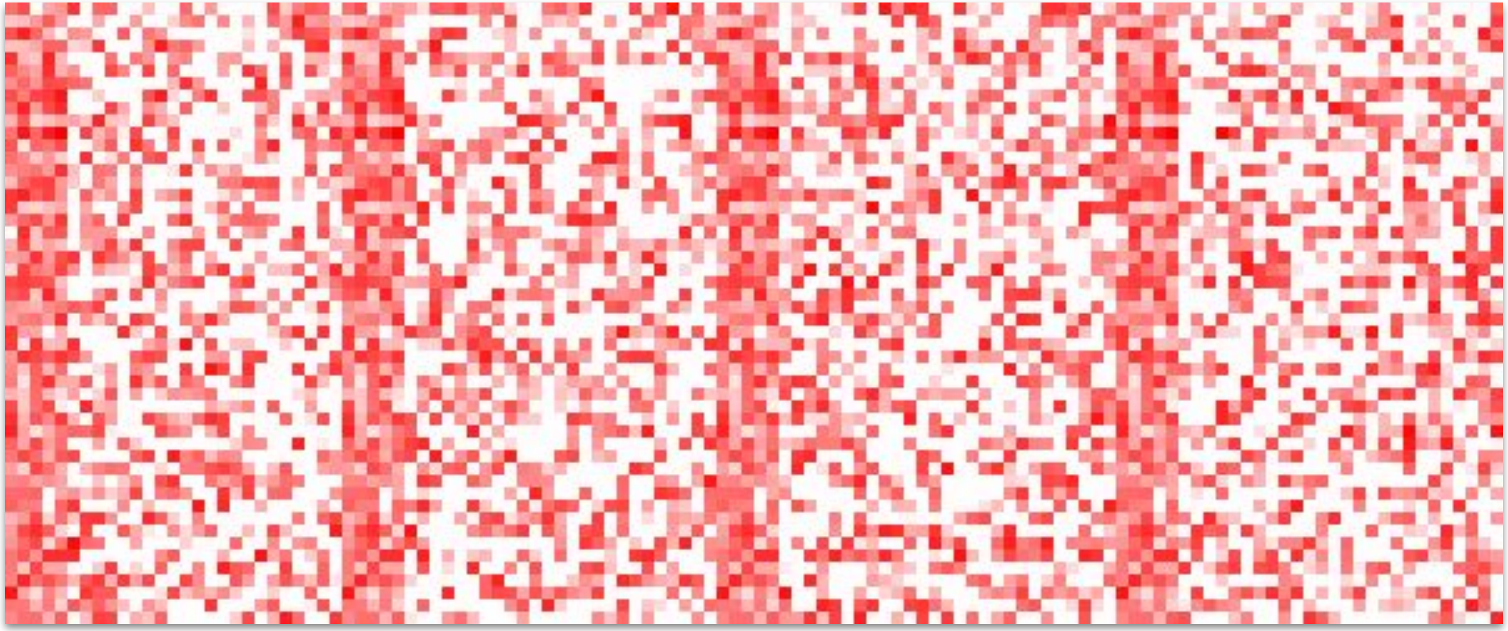
25%.



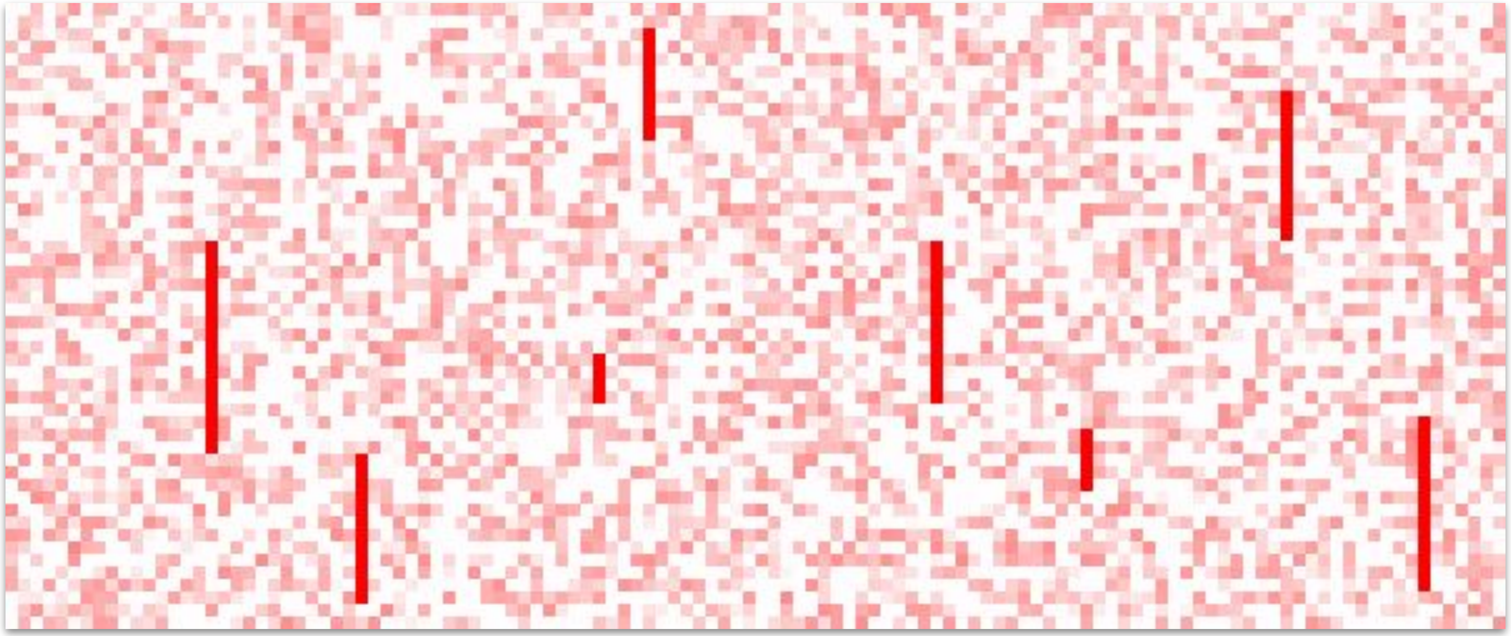
5%.



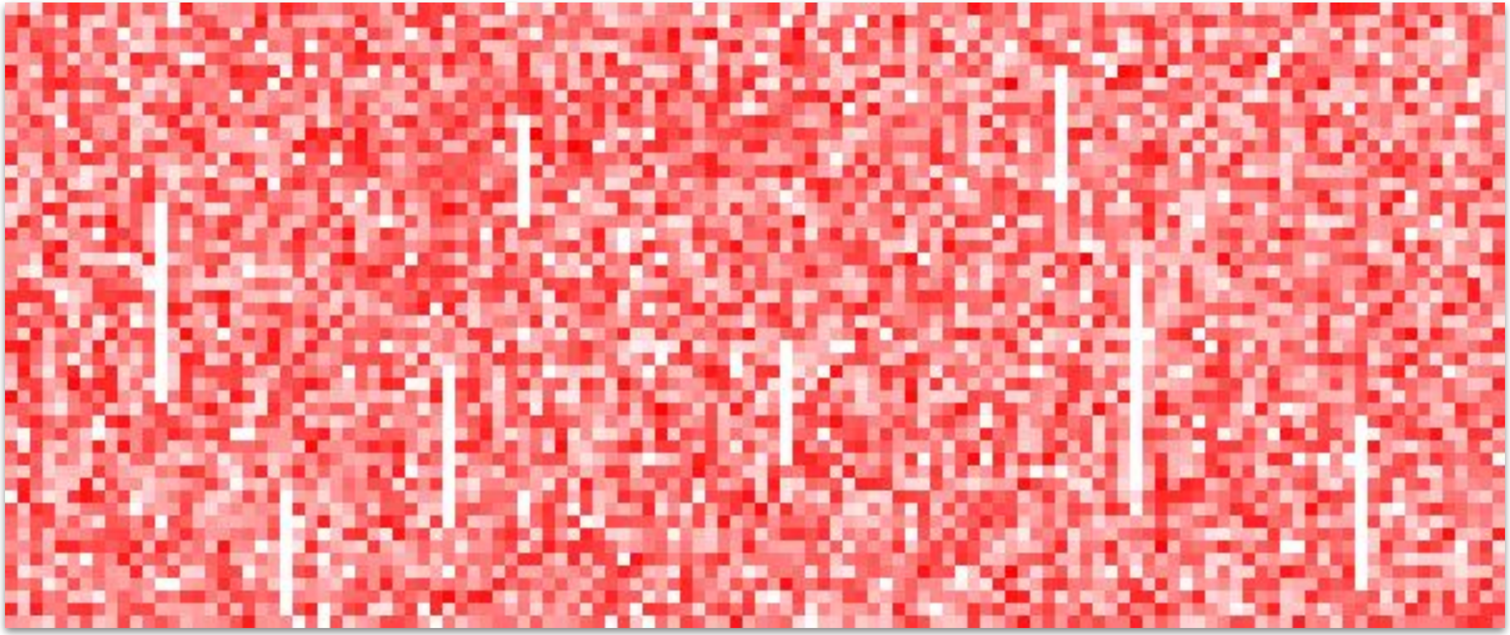
Load increasing.



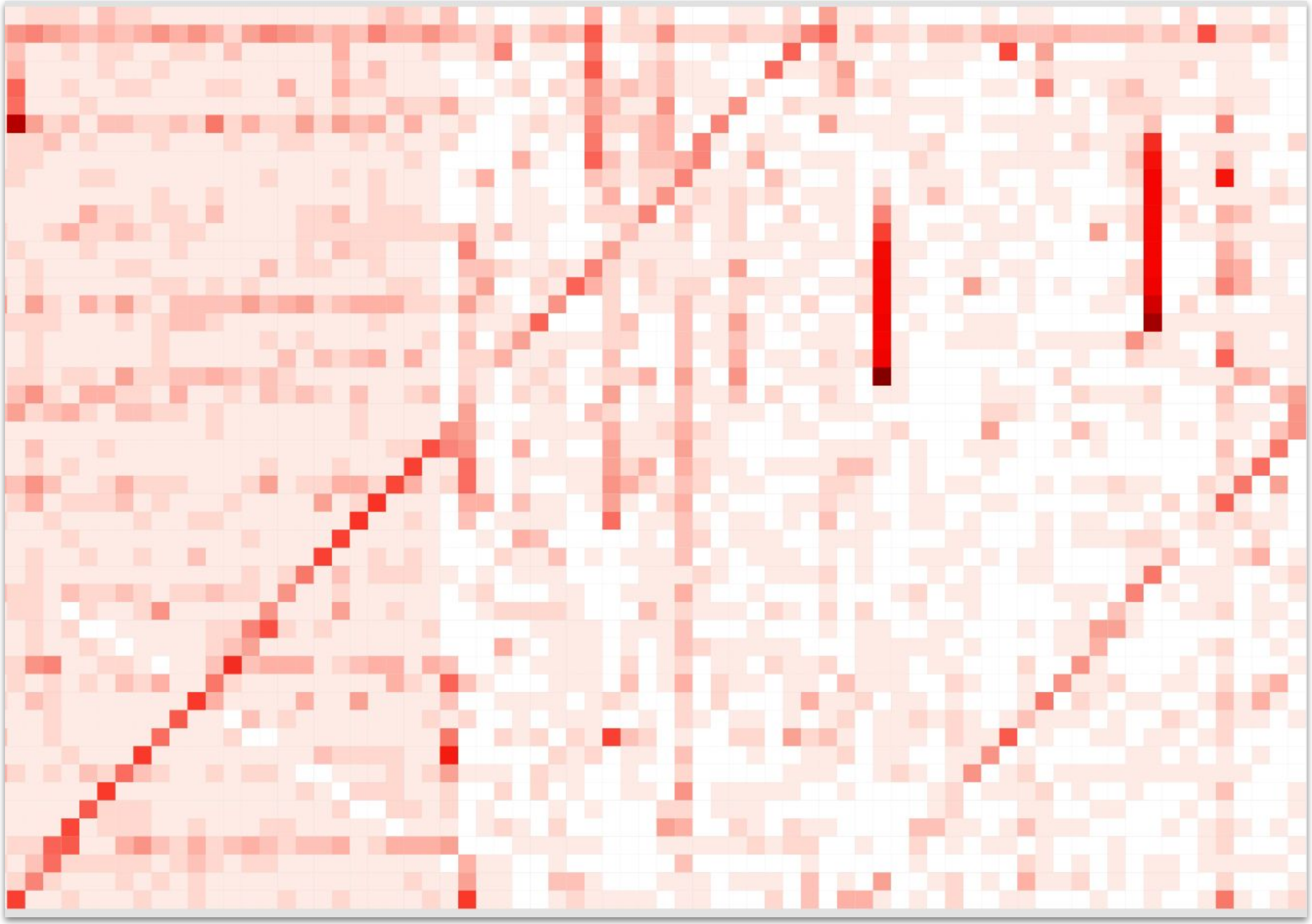
Variable load.



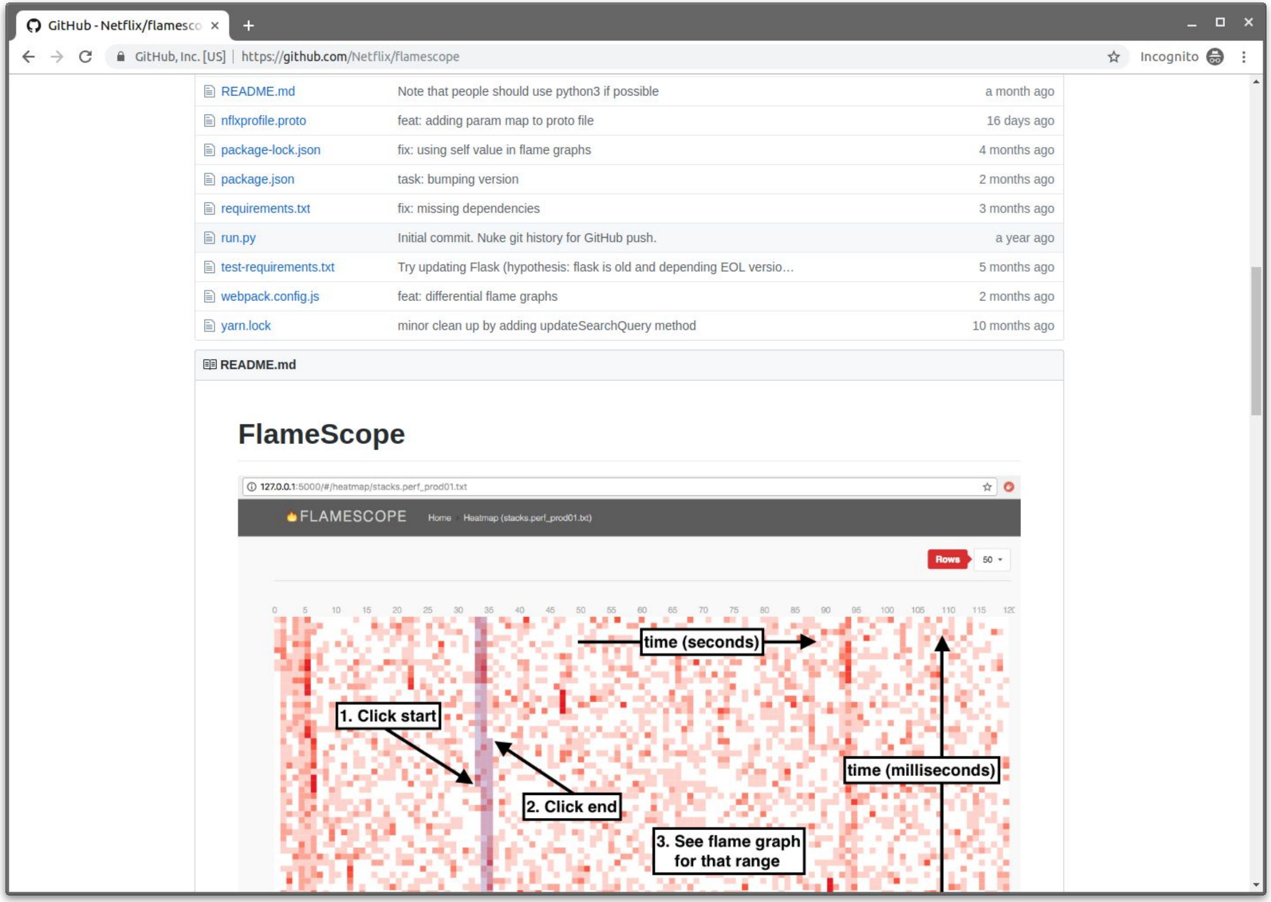
CPU perturbations.



CPU blocking.



It’s a simple visualization, but it allowed us to easily troubleshoot certain issues, and identify interesting patterns.



GitHub - Netflix/flamesco

GitHub, Inc. [US] | https://github.com/Netflix/flamescope

Incognito

README.md	Note that people should use python3 if possible	a month ago
nflxprofile.proto	feat: adding param map to proto file	16 days ago
package-lock.json	fix: using self value in flame graphs	4 months ago
package.json	task: bumping version	2 months ago
requirements.txt	fix: missing dependencies	3 months ago
run.py	Initial commit. Nuke git history for GitHub push.	a year ago
test-requirements.txt	Try updating Flask (hypothesis: flask is old and depending EOL versio…	5 months ago
webpack.config.js	feat: differential flame graphs	2 months ago
yarn.lock	minor clean up by adding updateSearchQuery method	10 months ago

README.md

FlameScope

127.0.0.1:5000/#/heatmap/stacks.perf_prod01.txt

https://github.com/Netflix/flamescope

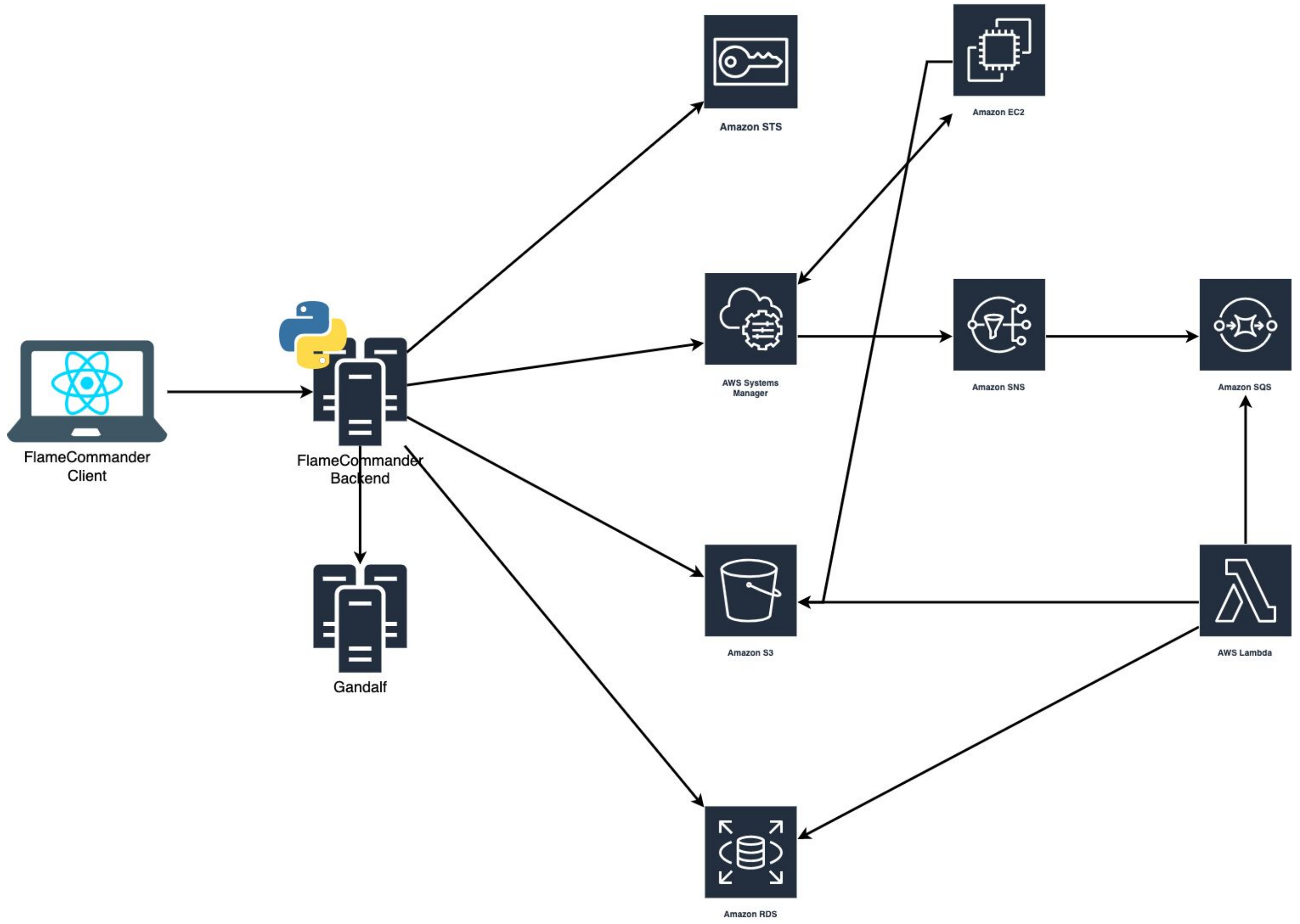
But FlameScope is not a full-fledged profiling solution ...

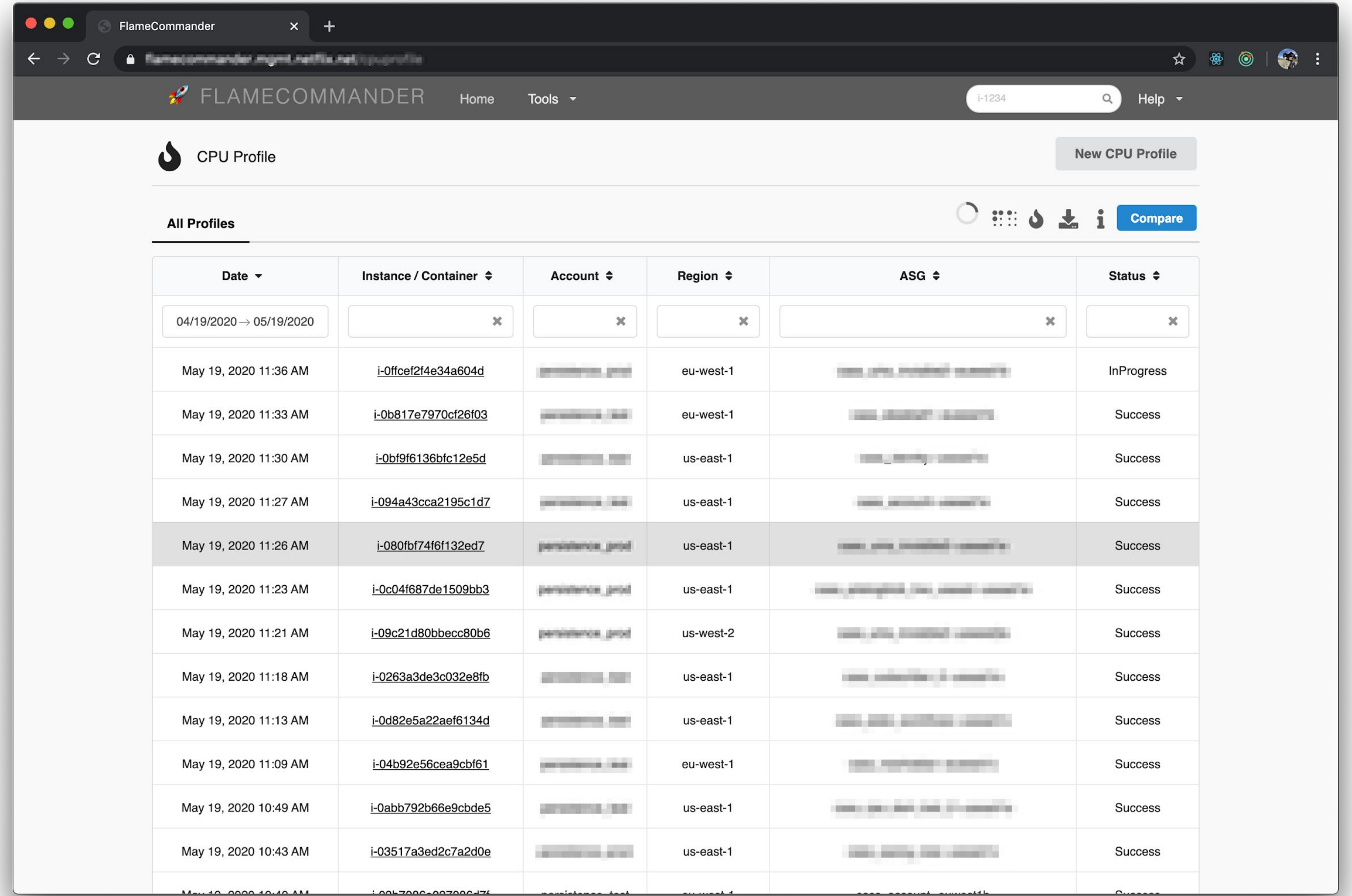
Wasn’t the lowest barrier of entry for engineers

We **scale** our efforts by creating easy-to-use **tools**

Why not have a centralized FlameScope?

FLAMECOMMANDER


Amazon STS
Amazon EC2
AWS Systems Manager
Amazon SNS
Amazon SQS
FlameCommander Client
FlameCommander Backend
Gandalf
Amazon S3
AWS Lambda
Amazon RDS



CPU Profile

All Profiles

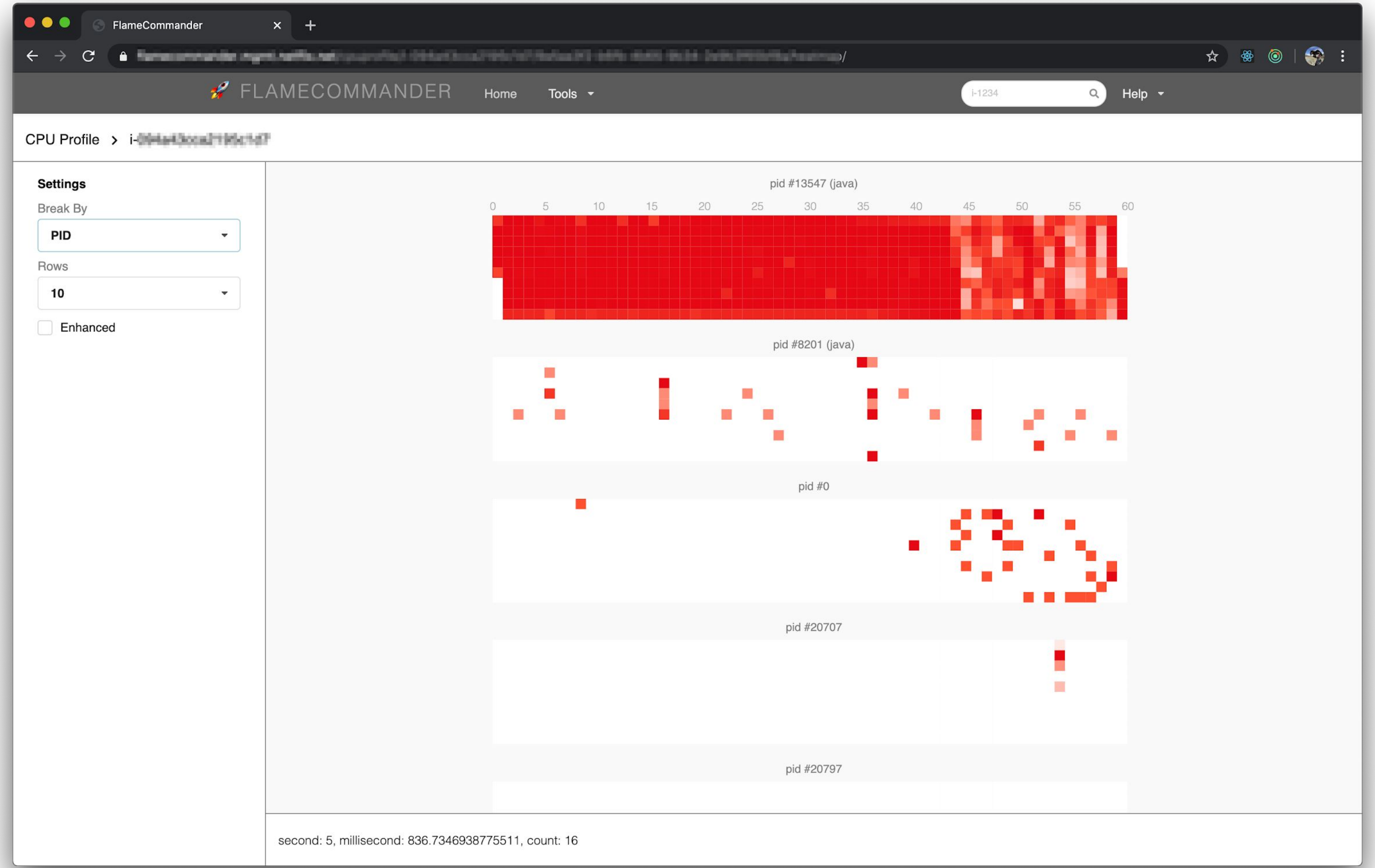
Date	Instance / Container	Account	Region	ASG	Status
04/19/2020 → 05/19/2020					
May 19, 2020 11:36 AM	i-0ffcef2f4e34a604d	[redacted]	eu-west-1	[redacted]	InProgress
May 19, 2020 11:33 AM	i-0b817e7970cf26f03	[redacted]	eu-west-1	[redacted]	Success
May 19, 2020 11:30 AM	i-0bf9f6136bfc12e5d	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:27 AM	i-094a43cca2195c1d7	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:26 AM	i-080fbf74f6f132ed7	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:23 AM	i-0c04f687de1509bb3	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:21 AM	i-09c21d80bbecc80b6	[redacted]	us-west-2	[redacted]	Success
May 19, 2020 11:18 AM	i-0263a3de3c032e8fb	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:13 AM	i-0d82e5a22aef6134d	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 11:09 AM	i-04b92e56cea9cbf61	[redacted]	eu-west-1	[redacted]	Success
May 19, 2020 10:49 AM	i-0abb792b66e9cbde5	[redacted]	us-east-1	[redacted]	Success
May 19, 2020 10:43 AM	i-03517a3ed2c7a2d0e	[redacted]	us-east-1	[redacted]	Success

More profilers.

- Heapdumps.
- Memory allocation profiles.
- More variations of CPU profiles.
- Off-CPU profiles.
- Adding more BPF-based tools.
- And a *bpftrace* interface.

More analysis options.

- Different stack parsers
  - Inverted merge
  - Package/module name
  - Demangle for different programming languages
- Break profiles by PID, TID and CPU



Settings

Break By: PID

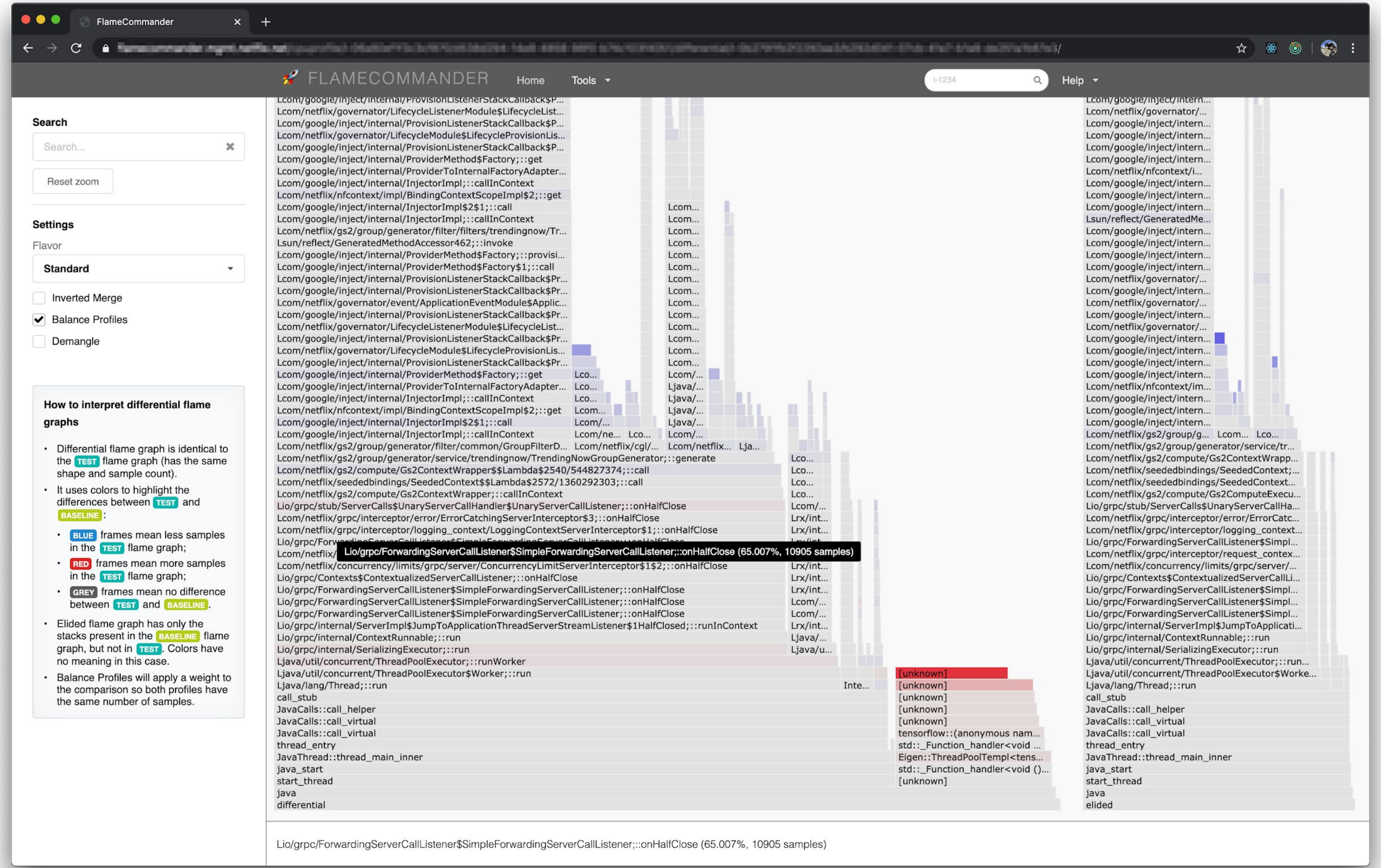
Rows: 10

Enhanced

second: 5, millisecond: 836.7346938775511, count: 16

More analysis options.

- Different stack parsers
  - Inverted merge
  - Package/module name
  - Demangle for different programming languages
- Break profiles by PID, TID and CPU
- Differential flame graphs



Search

Search...

Reset zoom

Settings

Flavor

Standard

[ ] Inverted Merge
[✓] Balance Profiles
[ ] Demangle

How to interpret differential flame graphs

- Differential flame graph is identical to the TEST flame graph (has the same shape and sample count).
- It uses colors to highlight the differences between TEST and BASELINE:
  - BLUE frames mean less samples in the TEST flame graph;
  - RED frames mean more samples in the TEST flame graph;
  - GREY frames mean no difference between TEST and BASELINE.
- Elided flame graph has only the stacks present in the BASELINE flame graph, but not in TEST. Colors have no meaning in this case.
- Balance Profiles will apply a weight to the comparison so both profiles have the same number of samples.

```
Lcom/google/inject/internal/ProvisionListenerStackCallback$P...
Lcom/netflix/governator/LifecycleListenerModule$LifecycleList...
Lcom/google/inject/internal/ProvisionListenerStackCallback$P...
Lcom/netflix/governator/LifecycleModule$LifecycleProvisionLis...
Lcom/google/inject/internal/ProvisionListenerStackCallback$P...
Lcom/google/inject/internal/ProviderMethod$Factory;::get
Lcom/google/inject/internal/ProviderToInternalFactoryAdapter...
Lcom/google/inject/internal/InjectorImpl;::callInContext
Lcom/netflix/nfcontext/impl/BindingContextScopeImpl$2;::get
Lcom/google/inject/internal/InjectorImpl$2$1;::call
Lcom/google/inject/internal/InjectorImpl;::callInContext
Lcom/netflix/gs2/group/generator/filter/filters/trendingnow/Tr...
Lsun/reflect/GeneratedMethodAccessor462;::invoke
Lcom/google/inject/internal/ProviderMethod$Factory;::provisi...
Lcom/google/inject/internal/ProviderMethod$Factory$1;::call
Lcom/google/inject/internal/ProvisionListenerStackCallback$Pr...
Lcom/google/inject/internal/ProvisionListenerStackCallback$Pr...
Lcom/netflix/governator/event/ApplicationEventModule$Applic...
Lcom/google/inject/internal/ProvisionListenerStackCallback$Pr...
Lcom/netflix/governator/LifecycleListenerModule$LifecycleList...
Lcom/google/inject/internal/ProvisionListenerStackCallback$Pr...
Lcom/netflix/governator/LifecycleModule$LifecycleProvisionLis...
Lcom/google/inject/internal/ProvisionListenerStackCallback$Pr...
Lcom/google/inject/internal/ProviderMethod$Factory;::get
Lcom/google/inject/internal/ProviderToInternalFactoryAdapter...
Lcom/google/inject/internal/InjectorImpl;::callInContext
Lcom/netflix/nfcontext/impl/BindingContextScopeImpl$2;::get
Lcom/google/inject/internal/InjectorImpl$2$1;::call
Lcom/google/inject/internal/InjectorImpl;::callInContext
Lcom/netflix/gs2/group/generator/filter/common/GroupFilterD...
Lcom/netflix/gs2/group/generator/service/trendingnow/TrendingNowGroupGenerator;::generate
Lcom/netflix/gs2/compute/Gs2ContextWrapper$$Lambda$2540/544827374;::call
Lcom/netflix/seededbindings/SeededContext$$Lambda$2572/1360292303;::call
Lcom/netflix/gs2/compute/Gs2ContextWrapper;::callInContext
Lio/grpc/stub/ServerCalls$UnaryServerCallHandler$UnaryServerCallListener;::onHalfClose
Lcom/netflix/grpc/interceptor/error/ErrorCatchingServerInterceptor$3;::onHalfClose
Lcom/netflix/grpc/interceptor/logging_context/LoggingContextServerInterceptor$1;::onHalfClose
Lio/grpc/Forw
Lcom/netflix/
Lcom/netflix/concurrency/limits/grpc/server/ConcurrencyLimitServerInterceptor$1$2;::onHalfClose
Lio/grpc/Contexts$ContextualizedServerCallListener;::onHalfClose
Lio/grpc/ForwardingServerCallListener$SimpleForwardingServerCallListener;::onHalfClose
Lio/grpc/ForwardingServerCallListener$SimpleForwardingServerCallListener;::onHalfClose
Lio/grpc/ForwardingServerCallListener$SimpleForwardingServerCallListener;::onHalfClose
Lio/grpc/internal/ServerImpl$JumpToApplicationThreadServerStreamListener$1HalfClosed;::runInContext
Lio/grpc/internal/ContextRunnable;::run
Lio/grpc/internal/SerializingExecutor;::run
Ljava/util/concurrent/ThreadPoolExecutor;::runWorker
Ljava/util/concurrent/ThreadPoolExecutor$Worker;::run
Ljava/lang/Thread;::run
call_stub
JavaCalls::call_helper
JavaCalls::call_virtual
JavaCalls::call_virtual
thread_entry
JavaThread::thread_main_inner
java_start
start_thread
java
differential
```

```
[unknown]
[unknown]
[unknown]
[unknown]
[unknown]
tensorflow::(anonymous nam...
std::_Function_handler<void ...
Eigen::ThreadPoolTempl<tens...
std::_Function_handler<void ()...
[unknown]
```

```
Lcom/google/inject/intern...
Lcom/netflix/governator/...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/netflix/nfcontext/i...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lsun/reflect/GeneratedMe...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/netflix/governator/...
Lcom/google/inject/intern...
Lcom/netflix/governator/...
Lcom/google/inject/intern...
Lcom/netflix/governator/...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/netflix/nfcontext/im...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/google/inject/intern...
Lcom/netflix/gs2/group/g...
Lcom/netflix/gs2/group/generator/service/tr...
Lcom/netflix/gs2/compute/Gs2ContextWrapp...
Lcom/netflix/seededbindings/SeededContext;:...
Lcom/netflix/seededbindings/SeededContext...
Lcom/netflix/gs2/compute/Gs2ComputeExecu...
Lio/grpc/stub/ServerCalls$UnaryServerCallHa...
Lcom/netflix/grpc/interceptor/error/ErrorCatc...
Lcom/netflix/grpc/interceptor/logging_context...
Lio/grpc/ForwardingServerCallListener$Simpl...
Lcom/netflix/grpc/interceptor/request_contex...
Lcom/netflix/concurrency/limits/grpc/server/...
Lio/grpc/Contexts$ContextualizedServerCallLi...
Lio/grpc/ForwardingServerCallListener$Simpl...
Lio/grpc/ForwardingServerCallListener$Simpl...
Lio/grpc/ForwardingServerCallListener$Simpl...
Lio/grpc/internal/ServerImpl$JumpToApplicati...
Lio/grpc/internal/ContextRunnable;::run
Lio/grpc/internal/SerializingExecutor;::run
Ljava/util/concurrent/ThreadPoolExecutor;::run...
Ljava/util/concurrent/ThreadPoolExecutor$Worke...
Ljava/lang/Thread;::run
call_stub
JavaCalls::call_helper
JavaCalls::call_virtual
JavaCalls::call_virtual
thread_entry
JavaThread::thread_main_inner
java_start
start_thread
java
elided
```

Lio/grpc/ForwardingServerCallListener$SimpleForwardingServerCallListener;::onHalfClose (65.007%, 10905 samples)

Lio/grpc/ForwardingServerCallListener$SimpleForwardingServerCallListener;::onHalfClose (65.007%, 10905 samples)

More analysis options.

- Different stack parsers
    - Inverted merge
    - Package/module name
    - Demangle for different programming languages
- Break profiles by PID, TID and CPU
- Differential flame graphs
- Working on middle-out merge
- Working on cloud-wide analysis

AUTOMATE
ALL THE THINGS
memegenerator.net

Takeaways.

- Don’t stick with line charts and tables for everything.
- Focus on lowering the barrier of entry.
- Centralized profiling solution helped with discoverability.
- All profiles are in the same place.
- Development cycle is faster.
- Automation is key to doing more.

Thank you.

Martin Spier
martinspier.io
@spiermar

NETFLIX