# Trace-based Analysis of Network Servers

**Nik Sultana*, Achala Rao*, Zihao Jin†, Pardis Pashakhanloo*, Henry Zhu*, Vinod Yegneswaran‡, Boon Thau Loo***

*University of Pennsylvania, †Tsinghua University, ‡SRI

**Presented at CNSM'19. Halifax, Canada**

## Problem

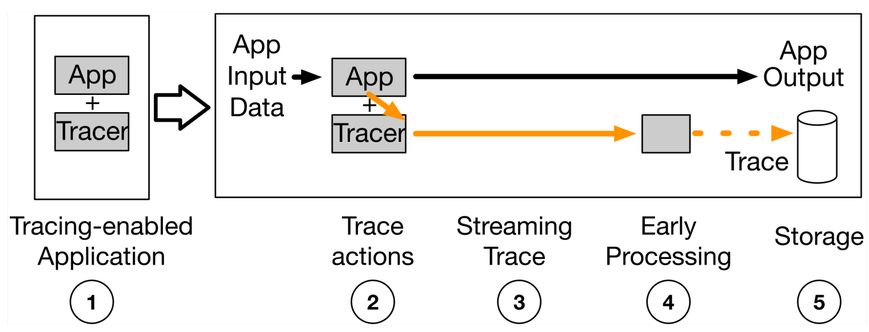
- Network servers -- for HTTP, FTP, etc -- are complex, multi-user systems.
- This complicates analysing their runtime, in-deployment behaviour, yet they are performance- and security-critical systems.
- How can we better analyse and understand their behaviour, to better detect and fix problems?

## Our solution: Flowdar

- Configurable tracing using custom + existing tools.
- Trace simplification, in both application-agnostic and application-specific ways.
- Trace visualisation.


## Flowdar Design

1. Patch application to produce traces at configurable detail.
2. Run workloads on application to generate traces.
3. Traces are put through an in-memory pipeline to reduce blocking before storage.
4. Light preprocessing is done to eliminate unnecessary details.
5. Trace is stored and processed to filter details and analysed in an application-specific way to demultiplex different users' sessions, trace activities across threads, etc. Traces are vastly compressed. We developed a rich visualisation to make traces more understandable.




## Example 1: Denial-of-Service analysis

Flowdar can automatically compare + simplify DoS and non-DoS workloads to find out which parts of the application are being affected.

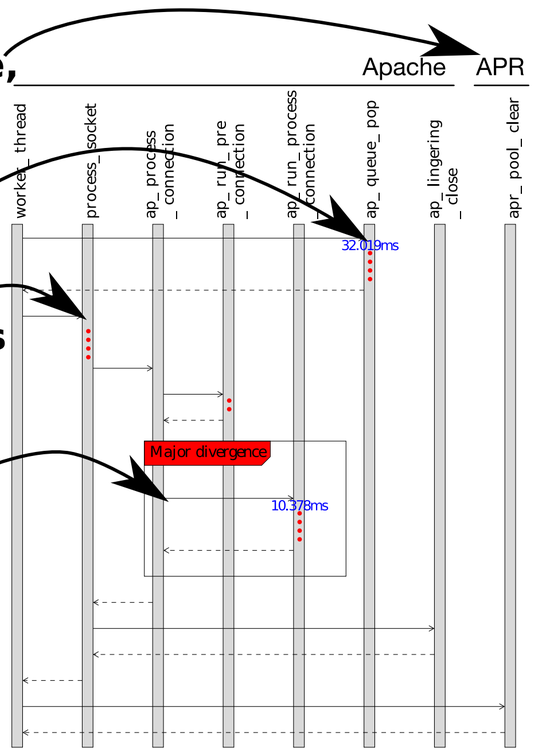
We applied this to the Apache Web Server. Further, this is visualised as a sequence diagram by our tools:

**Apache Portable Runtime, an Apache dependency.**

**Function call's duration**

**Each red dot represents 500us. Excess of 4 dots is shown numerically.**

**400x average difference in duration between DoS and non-DoS workloads.**

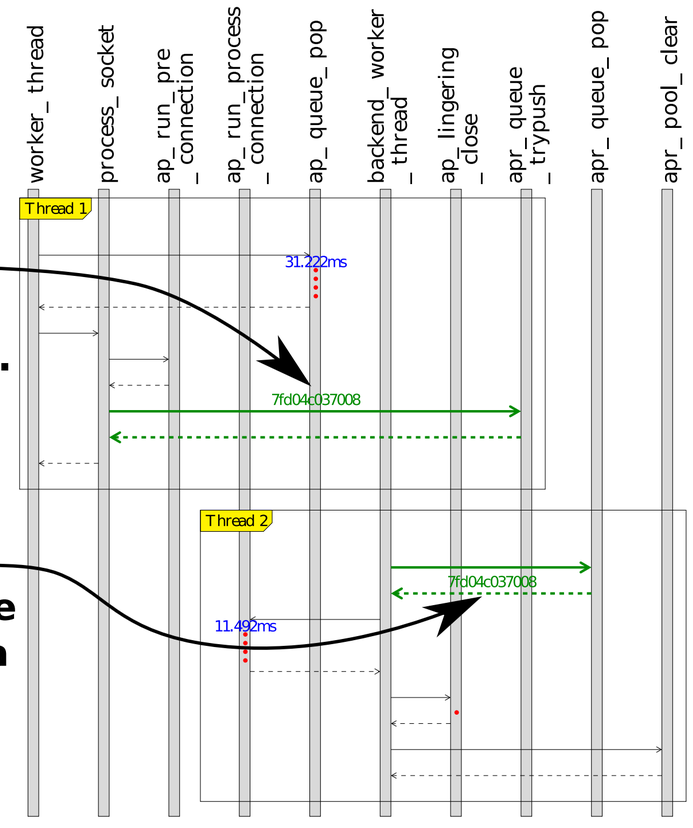



## Example 2: Thread coordination

To mitigate DoS we pipelined Apache's Worker threads to have different pools of worker threads. Visualisation shows hand-over of the connection record between threads in this pipeline.

**Memory address of connection record queued by first thread.**

**We can observe the processing of the same connection record by a downstream thread.**




**Full source-code + documentation + examples https://gitlab.com/DeDos/flowdar**

We thank Bob DiMaiolo and John Frommeyer for prototyping and systems help. This work is supported in part by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-16-C-0056 and HR0011-17-C-0047.