Failure Sketches: A Better Way to Debug

Baris Kasikci, Cristiano Pereira, Gilles Pokam, Benjamin Schubert, Madan Musuvathi, George Candea
Failure and Root Cause

- Failure
  - Violation of a program specification
  - Memory errors, hangs, etc

- Root cause
  - “The real reason” behind the failure
  - When removed from the program, the failure does not recur
Debugging In-Production Software Failures Today
Debugging In-Production Software Failures Today
Debugging In-Production Software Failures Today

```c
#0  0x00007f51abaee820b in raise (sig=11) at ../nptl/sysdeps/unix/sysv/linux/pt-raise.c:37
#1  0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=82) at mod_log_config.c:1368
#2  0x000000000042d10d in config_log_transaction (r=0x7f51a40053d0, cli=0x20b6b450, default_format=0x200e370) at mod_log_config.c:930
#3  0x000000000042d2d6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4  0x000000000046cb2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
#5  0x0000000000403681 in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#6  0x0000000000465cdd in ap_run_process_connection (r=0x7f51a40053d0) at connection.c:85
#7  0x00000000004661f5 in ap_process_connection (c=0x7f519c000b68, csd=0x7f519c000a20) at connection.c:211
#8  0x0000000000451ba0 in process_socket (p=0x7f519c0009b8, sock=0x7f519c000a20, my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51a4001348) at worker.c:632
#10 0x0000000000451221 in worker_thread (thd=0x210fa90, dummy=0x7f51a4002000) at worker.c:946
#11 0x0000000000451221 in worker_thread (opaque=0x210fa90) at thread.c:1107
#12 0x0000000000451221 in start_thread (arg=0x7f51a400e700) at pthread_create.c:312
#13 0x000000000044b4f7 in clone () at ../sysdeps/unix/sysv/linux/clone.c:111
```
Debugging In-Production Software Failures Today

```c
#0 0x000007f51aabaeb20b in raise (sig=11) at ../nptl/sysdeps/unix/sysv/linux/pt-raise.c:37
#1 0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=82) at mod_log_config.c:1368
#2 0x000000000042e710 in config_log_transaction (r=0x7f51a40053d0, cl=0x20ebd450, default_format=0x20ee370) at mod_log_config.c:930
#3 0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4 0x000000000042aad6 in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
#5 0x000000000042aad6 in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#6 0x000000000042ead1 in ap_process_http_transaction (r=0x7f51a40053d0) at http_core.c:293
#7 0x000000000042ead1 in ap_run_process_connection (r=0x7f51a40053d0) at connection.c:85
#8 0x000000000042ead1 in ap_process_connection (c=0x7f519c000b68, csd=0x7f519c000a20) at connection.c:211
#9 0x000000000042ead1 in process_socket (p=0x7f519c0009b8, sock=0x7f519c000a20, my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51a4001348) at worker.c:632
#10 0x000000000042ead1 in worker_thread (thd=0x210fa90, dummy=0x7f51a40008d0) at worker.c:846
#11 0x000000000042ead1 in ap_process_request (opaque=0x210fa90) at ap_process.c:107
#12 0x000000000042ead1 in start_thread (arg=0x7f51a8e6f70) at pthread_create.c:312
#13 0x000000000042ead1 in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:111
```

Understand root cause
Debugging In-Production Software Failures Today

Understand root cause

Reproduce the problem

```
# 0x00007f51abaeb20b in raise (sig=11) at ../nptl/sysdeps/unix/sysv/linux/pt-raise.c:37
# 1 0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0,
  str=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=82) at mod_log_config.c:1368
# 2 0x000000000042d10d in config_log_transaction
  (r=0x7f51a40053d0, cli=0x20eeb4d0, default_format=0x20ee370) at mod_log_config.c:930
# 3 0x000000000042d0ad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
# 4 0x000000000042d0b2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:11563
# 5 0x000000000042d0b1e81 in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
# 6 0x000000000042d0b29a in ap_process_http_connection (r=0x7f51b0000b68) at http_core.c:293
# 7 0x000000000042d0b06d in ap_run_process_connection (r=0x7f51b0000b68) at connection.c:85
# 8 0x000000000042d0b06f5 in ap_process_connection (r=0x7f51b0000b68, c=0x7f519c00a20) at
  connection.c:211
# 9 0x000000000042f51b51ba0 in process_socket (p=0x7f519c0000b8, sock=0x7f519c0000a20,
  my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51a4001348) at worker.c:632
#10 0x000000000042f51b2221 in worker_thread (thd=0x2010fa90, dummy=0x7f51a400a00d) at worker.c:946
#11 0x000000000042f51a87c555 in dummy_worker (opaque=0x2010fa90) at thread.c:127
#12 0x000000000042f51ab0182 in start_thread (arg=0x7f51a8e700) at pthread_create.c:312
#13 0x000000000042f51ab0d47d in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.c:111
```
Debugging In-Production Software Failures Today

Understand root cause

Reproduce the problem

```c
#0 0x00007f51aba4820b in raise (sig=11) at ../opt1/sysdeps/unix/sysv/linux/pt-raise.c:37
#1 0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, strs1=0x7f51a40035e8, nelts=14, len=82) at mod_log_config.c:1368
#2 0x000000000042b10d in config_log_transaction (r=0x7f51a40053d0, cls=0x20b9d50, default_format=0x20ee370) at mod_log_config.c:930
#3 0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4 0x000000000042aad6 in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1163
#5 0x000000000042aad6 in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#6 0x000000000042aad6 in ap_process_http_connection (c=0x7f519c000b68) at http_core.c:293
#7 0x000000000042aad6 in ap_run_process_connection (c=0x7f519c000b68) at connection.c:85
#8 0x000000000042aad6 in ap_process_connection (c=0x7f519c000b68, csd=0x7f519c000a20) at connection.c:211
#9 0x00000000004051ba0 in process_socket (p=0x7f519c000b8, sock=0x7f519c000a20, my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51a4003148) at worker.c:632
#10 0x00000000004051ba0 in worker_thread (thd=0x210fa90, dummy=0x7f51a4008800) at worker.c:946
#11 0x00000000004051ba0 in dummy_worker (opaque=0x210fa90) at thread.c:127
#12 0x00000000004051ba0 in start_thread (arg=0x7f51a8ef700) at pthread_create.c:312
#13 0x00000000004051ba0 in thread_create (arg=0x7f51a8ef700) at pthread_create.c:312
```

Understand root cause

Reproduce the problem
Tackling the Debugging Challenge

• Record/replay

• Special runtime support\textsuperscript{1}
  • VM checkpointing

• Custom hardware\textsuperscript{2}
  • Not widely available

\textsuperscript{1} J. Tucek et al., Triage: Diagnosing Production Run Failures at the User's Site, SOSP 2007
\textsuperscript{2} G. Pokam et al., QuickRec: prototyping an intel architecture extension for record and replay of multithreaded programs, ISCA 2013
Tackling the Debugging Challenge

- Record/replay
- Special runtime support\(^1\)
  - VM checkpointing
- Custom hardware\(^2\)
  - Not widely available

Existing tools don’t help debugging in-production failures\(^3\)

---

\(^1\) J. Tucek et al., Triage: Diagnosing Production Run Failures at the User's Site, SOSP 2007

\(^2\) G. Pokam et al., QuickRec: prototyping an intel architecture extension for record and replay of multithreaded programs, ISCA 2013

\(^3\) C. Sadowski et al., How developers use data race detection tools, Workshop on Evaluation and Usability of Programming Languages and Tools 2014
Debugging In-Production Software Failures Today

Understand root cause

Reproduce the problem

---

0x00007ff51abae820b in raise (sig=11) at ../sysdeps/unix/sysv/linux/pt-raise.c:37
#1 0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, nels=14, len=82) at mod_log_config.c:1368
#2 0x000000000042610d in config_log_transaction (r=0x7f51a40053d0, cls=0x20b9d50, default_format=0x20ee370) at mod_log_config.c:930
#3 0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4 0x000000000046cb2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
#5 0x000000000042e9da in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#6 0x000000000042b10d in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, nels=14, len=82) at mod_log_config.c:1368
#7 0x00000000004661f5 in ap_process_connection (c=0x7f519c000b68) at connection.c:211
#8 0x0000000000451ba0 in process_socket (p=0x7f519c0009b8, sock=0x7f519c0009a0, my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51a4001348) at worker.c:632
#9 0x0000000000451221 in worker_thread (thd=0x210fa90, dummy=0x7f51a40008c0) at worker.c:196
#10 0x0000000000450221 in worker_thread (thd=0x210fa90, dummy=0x7f51a40008c0) at worker.c:196
#11 0x00007ff51abed8f in dumby_worker (Opaque=0x210fa90) at thread.c:127
#12 0x00007ff51abed82 in start_thread (arg=0x7f51a4007f00) at pthread_create.c:312
#13 0x00007ff51abed7d in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:111
Failure Sketch

Thread 1

```c
main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}
```

Thread 2

```c
cons(queue* f) {
    ...
    mutex_unlock(f->mut);
    f->mut = NULL;
    ...
}
```

Time

Thread 1

1. main() {
2.     queue* f = init(size);
3.     create_thread(cons, f);
4.     ...
5.     free(f->mut);
6.     f->mut = NULL;
7.     ...
8. }

Thread 2

1. cons(queue* f) {
2.     ...
3.     mutex_unlock(f->mut);
4.     f->mut = NULL;
5.     ...
6. }

Root cause

Failure: segmentation fault
Failure Sketch

Time

Thread 1

1 main() {
2 queue* f = init(size);
3 create_thread(cons, f);
4 ...
5 free(f->mut);
6 \textbf{f->mut = NULL;}
7 ...
8 }

Thread 2

1 cons(queue* f) {
2 ...
3 \textbf{mutex_unlock(f->mut);} \\
4 ...
5 }

\textbf{Failure: segmentation fault}
Failure Sketch

```c
main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}

cons(queue* f) {
    ...
    mutex_unlock(f->mut);
}
```

Thread 1

Thread 2

Time

1 main() {
2     queue* f = init(size);
3     create_thread(cons, f);
4     ...
5     free(f->mut);
6     f->mut = NULL;  
7     ...
8 }

Root cause

Failure: segmentation fault
Failure Sketch Use Case

Understand root cause

Reproduce the problem

0x00007f51aeb3828b in raise (sig=11) at ./..:/sysdeps/unix/sysv/linux/pt-raise.c:37
1 0x0000000000042d89 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b8eb0, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
2 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
3 0x0000000000042d289 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b9d50, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
4 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
5 0x0000000000042d289 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b9d50, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
6 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
7 0x0000000000042d289 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b9d50, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
8 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
9 0x0000000000042d289 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b9d50, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
10 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
11 0x0000000000042d289 in ap_buffered_log_writer (r=0x7f51a4005930, handle=0x20b9d50, str=0x7f51a4003358, strl=14, len=62) at mod_log_config.c:151
12 0x0000000000042d10d in config_log_transaction (r=0x7f51a4005930, cls=0x20b9d50, default_format=0x20b9370) at mod_log_config.c:930
13 0x00007f51aeb3828b in raise (sig=11) at ./..:/sysdeps/unix/sysv/linux/pt-raise.c:37
Failure Sketch Use Case

Understand root cause

Reproduce the problem

```
#0  0x00007f51a6ae820b in raise (sig=11) at ../../sysdeps/unix/sysv/linux/pt-raise.c:37
#1  0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, strs=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=82) at mod_log_config.c:1368
#2  0x000000000042e210 in config_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:930
#3  0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4  0x000000000046cb2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
#5  0x000000000042e9da in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#6  0x000000000042e9da in ap_process_request (r=0x7f51a40053d0) at http_request.c:312
#7  0x00007f51ac87c555 in dummy_worker (opaque=0x210fa90) at thread.c:127
#8  0x00007f51ab4e5a0 in process_socket (p=0x7f51ab0078b8, sock=0x7f51ab0070a0, my_child_num=0, my_thread_num=0, bucket_alloc=0x7f51400134b) at worker.c:632
#10 0x0000000000451221 in worker_thread (thd=0x210fa90, dummy=0x7f51400080c0) at worker.c:946
#11 0x00007f51ab7555 in dummy_worker (opaque=0x210fa90) at thread.c:127
#12 0x00007f51ab4e5a2 in start_thread (arg=0x7f51ab0070a0) at pthread_create.c:312
#13 0x00007f51ab8047d in clone () at ../../sysdeps/unix/sysv/linux/x86_64/clone.S:111
```
Failure Sketch Use Case

Understand root cause

Reproduce the problem

```
#0  0x00007f51a0ee820b in raise (sig=11) at ../nptl/sysdeps/unix/sysv/linux/pt-raise.c:37
#1  0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, stra=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=32) at mod_log_config.c:1368
#2  0x000000000042621d in config_log_transaction (r=0x7f51a40053d0, cls=0x20b9d50, default_format=0x20ee370) at mod_log_config.c:930
#3  0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:950
#4  0x000000000046cb2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
#5  0x000000000042b10d in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, stra=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=32) at mod_log_config.c:1368
#6  0x000000000042d289 in ap_buffered_log_writer (r=0x7f51a40053d0, handle=0x20eeba0, stra=0x7f51a4003578, strl=0x7f51a40035e8, nelts=14, len=32) at mod_log_config.c:1368
#7  0x000000000042aad6 in multi_log_transaction (r=0x7f51a40053d0) at mod_log_config.c:930
#8  0x000000000046cb2d in ap_run_log_transaction (r=0x7f51a40053d0) at protocol.c:1563
```

Runtime traces
Failure Sketch Use Case

Understand root cause

Reproduce the problem

Time

Thread 1
1 main() {
2 queue* f = init(size);
3 create_thread(cons, f);
4 ...
5 free(f->mut);
6 f->mut = NULL;
7 ...
8 }

Thread 2
1 cons(queue* f) {
2 ...
3 mutex_unlock(f->mut);
4 ...
5 ...
6 ...}

Failure: segmentation fault

Runtime traces
Failure Sketch Use Case

Understand root cause

```
main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}
cons(queue* f) {
    ...
    mutex_unlock(f->mut);
}
```

Time

Thread 1

Thread 2

Failure: segmentation fault
Failure Sketch Use Case

```c
main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}
cons(queue* f) {
    ...
    mutex_unlock(f->mut);
    ...
}
```

Thread 1

Thread 2

Time

Runtime traces

Failure: segmentation fault
Failure Sketch Use Case

main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}

cons(queue* f) {
    ...
    mutex_unlock(f->mut);
    ...
}

Time

Thread 1

Thread 2

Runtime traces

Failure: segmentation fault
Research Challenges

• Hard-to-reproduce failures
  • Recur only a few times in production

• Accuracy of failure sketches
  • No extraneous elements in the failure sketch

• Latency of failure sketch computation
  • Developers can’t wait forever for failure sketches
System Architecture

```c
main() {
    queue* f = init(size);
    create_thread(cons, f);
    ...
    free(f->mut);
    f->mut = NULL;
    ...
}

cons(queue* f) {
    ...
    mutex_unlock(f->mut);
}
```

Time

Thread 1

Thread 2

Runtime traces

Failure: segmentation fault
System Architecture

Program P (source)

Failure report (core dump, stack trace, etc)

Server

1
Program P (source)

Failure report (core dump, stack trace, etc)

Static Analyzer

Server

Static slice

1. queue* f = init(size);
2. create_thread(cons, f);
3. free(f->mut);
4. f->mut = NULL;
5. mutex_unlock(f->mut);
Program P (source)

Failure report (core dump, stack trace, etc)

Static Analyzer

Instrumentation

Tracking control and data flow
System Architecture

1. Program P (source)
2. Failure report (core dump, stack trace, etc)
3. Static Analyzer
4. Refinement with runtime traces
5. Instrumentation
6. Client
7. Tracking control and data flow
System Architecture

Failure Sketch

1. Program P (source)
2. Failure report (core dump, stack trace, etc)
3. Instrumentation
4. Failure Sketch Computation Engine

Server

Static Analyzer

Client

Tracking control and data flow

Thread 1

main()
\{ 
  queue* f = init(size);
  create_thread(con, f);
  ... 
  free(f->mut);
  f->mut = NULL;
  ...
\}

Thread 2

cons(queue* f) 
\{ 
  ... 
  mutex_unlock(f->mut);
\}

Failure: segmentation fault

Time

Program P

Failure Sketch

Thread 1

Thread 2
Intel Processor Trace (Intel PT)

- Control flow information
  - Compressed trace of branches taken (~1 bit per instruction)
  - Low overhead (~40% **full** tracing overhead)
Tracking Control Flow Using Intel PT
Tracking Control Flow Using Intel PT

Static Slice

Root cause

Failure
Tracking Control Flow Using Intel PT

Static Slice

------

Root cause

-----------

Failure

------

Tracking 1\textsuperscript{st} iteration
## Tracking Control Flow Using Intel PT

<table>
<thead>
<tr>
<th>Static Slice</th>
<th>Tracking 1\text{st} iteration</th>
<th>Tracking 2\text{nd} iteration</th>
<th>Tracking 3\text{rd} iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root cause</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tracking Control Flow Using Intel PT

<table>
<thead>
<tr>
<th>Static Slice</th>
<th>Tracking 1(^{\text{st}}) iteration</th>
<th>Tracking 2(^{\text{nd}}) iteration</th>
<th>Tracking 3(^{\text{rd}}) iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root cause</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Monitoring small portions of a slice works well because most failures have nearby root causes\(^1,2\)

\(^1\) W. Zhang et al., ConSeq: Detecting concurrency bugs through sequential errors. ASPLOS 2011
\(^2\) F. Qin et al., Rx: Treating bugs as allergies a safe method to survive software failures. SOSP 2005
Discussion

• Intrusiveness
  • Currently, we do static instrumentation
  • Dynamic instrumentation is less intrusive

• Privacy
  • Use anonymization
  • Forgo data monitoring when privacy requirements are very strict
Future Work

- Diagnosing performance problems
- Correlating control flow with slowdowns
- Speeding up program analysis
  - Use control flow information to tackle path explosion
- Using failure sketches for test case generation
• Failure sketches
  • Summary explaining failure root causes
• Application of hardware-based monitoring
  • Enabler for building failure sketches
• Many potential use cases