CUSTOMER SUCCESS STORY

FROM 3 WEEKS TO 15 MINUTES

How Palo Alto Networks resolved a long-standing customer-reported defect with Undo AI

100×	15 min	2	0
faster time to resolution	to root cause (vs ~3 weeks)	AI queries to full diagnosis	on-site visits needed

CONTEXT

Background

Quality is paramount to Palo Alto Networks (PAN). In their complex, large-scale deployments, issues occasionally surface in the field, and PAN prioritizes resolving them quickly.

Engineers use Undo as part of their debugging workflow for complex software issues in firewall products (physical units deployed on premise at customer sites). When a process crashes, the team typically receives a customer ticket with a backtrace and sometimes a core file.

Harshith, a software engineer on the debugging team, had been using Undo for some time. Before Undo, similar investigations often took around 3 weeks to resolve. Undo's time travel debugging capability (based on execution record/replay) reduced that to 1 or 2 days, significantly improving productivity. Undo's new AI-powered capability would reduce that timeframe even further.

THE PROBLEM

A bug that defied 3 previous attempts

A customer reported a process crash on a PAN firewall. The bug had been reported several times before, but the team had been unable to resolve it because they could not reliably reproduce it.

The crash was caused by a double-free memory corruption issue: a process was attempting to free the same block of memory twice, causing it to crash. While the team had a backtrace, these bugs are difficult to diagnose because a backtrace shows where a process crashed, not the sequence of events that caused it.

The challenge was determining which execution path led into the error-handling logic. A function contains multiple conditional branches, each capable of routing execution into an error-handling block. That error block in turn calls further functions and from a backtrace, you can see the call chain, but you cannot tell which of the several conditions triggered the error path in the first place. Even with GDB, the team could not determine which configuration state triggered the failure.

The team suspected a corner case in the configuration parsing logic. This time, however, the customer could reliably reproduce the crash by following a specific sequence of steps, creating an opportunity for full investigation.

SOLUTION

How it was solved — five steps

1 Bug received and assessed

Harshith reviewed the backtrace from the customer. Recognizing the pattern from previous reports, he decided to try a recording-based approach now that a repro was possible.

2 TAC team coached and recording taken

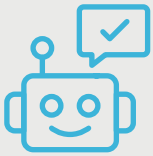
The TAC (Technial Assistance Center) engineer assigned to the customer hadn't used Undo before. Harshith spent around an hour coaching him using Undo's existing documentation. The process was straightforward: they attached Undo's LiveRecorder tool to the specific crashing process, the customer followed the steps to trigger the issue, and the recording was captured automatically. The whole recording took just 3–4 minutes. Due to time zone differences, it was ready for Harshith the next morning.

3 First AI query: finding the crash

With the recording in hand — and source code and symbols provided — Harshith used Undo's AI capability to connect to the agent, asking: "Find why this process is crashing." The AI navigated the full recording autonomously (treating it like an interactive debugging session) and queried through the history to retrieve the control flow and data flow of individual functions. Within 7.5 minutes it had identified which function was leading to the crash and given a strong direction on the root cause, but the exact trigger still needed pinning down.

4 Follow-up query: pinpointing the root cause

The first query showed where the crash occurred, but not what had triggered the jump into the error block, the critical missing piece that a backtrace alone can't answer. Harshith directed the AI towards the configuration parsing logic and asked a targeted follow-up. The AI traced the conditional path through the relevant code, examined the config values present at the time of execution, and identified the exact root cause: a specific corner-case configuration state that caused the process to enter the error block incorrectly, resulting in the double-free. It identified both where the first and second free occurred.



5 Workaround delivered, proper fix to follow

Harshith provided the customer with a workaround to unblock them immediately. The fix was later confirmed and committed with no further issues.

RESULTS

Impact at a glance

100× faster time-to-resolution

A long-standing bug was fully diagnosed in just 15 minutes — down from an estimated 3 weeks without Undo.

Customer unblocked quickly

A workaround was provided immediately, allowing the customer to continue while a proper fix was prepared and committed.

No onsite visit required

The recording was taken entirely remotely during a call with the customer. No travel, no delays, no access issues.

Repeatable process, low barrier

A TAC engineer with no prior Undo experience was up and running in around an hour, using existing documentation and brief coaching.

> *"Before Undo AI was introduced, I would have taken one to two days to debug things. With it, we got the exact root cause in 15 minutes."*

Harshith K
Software Engineer, Palo Alto Networks

LEARN MORE AT UNDO.IO