

A Primer in Root Cause Analysis

E-book by

Mark Henry

Senior System Engineer



FirstWave

A Primer in Root Cause Analysis

We've seen it time and time again, a ticket comes into the help desk, a customer is complaining about a slow application or poor voice quality during a call. We start digging into the problem, maybe pull some logs, grab some performance data from the NMS. Everything we find is inconclusive, and when we check back with the client the symptoms have passed. The problem is gone, and another ticket is in the queue, so we move on – no wiser as to what caused the issue – knowing that it will reappear.

The process of getting to the core, or root of a fault or problem is called Root Cause Analysis (RCA). Root Cause Analysis is not a single, stringent process, but rather a collection of steps, often organized specifically by type of problem or industry, which can be customized for a particular problem. When it comes to identifying the root cause of an IT-related event a combination of process-based and failure-based analysis can be employed. By applying an RCA process, and remediating issues to prevent their future occurrence, reactive engineering teams can be transformed into proactive ones that solve problems before they occur or escalate.



In this article I will attempt to outline a general process for troubleshooting network-related events, meaning those issues which directly impact the performance of a computer network or application resulting in a negative impact on user experience. While I will use FirstWave's Solutions in the examples, these steps can be applied to any collection of NMS tools.



The problem is gone, and another ticket is in the queue, so we move on – no wiser as to what caused the issue – knowing that it will reappear.

Table of Contents




An Introduction to Root Cause Analysis	4
Identifying the Event or Problem	5
Establish a Timeline	7
Root Causes vs. Causal Factors	9
Closing out the Event	11
About the Author	12





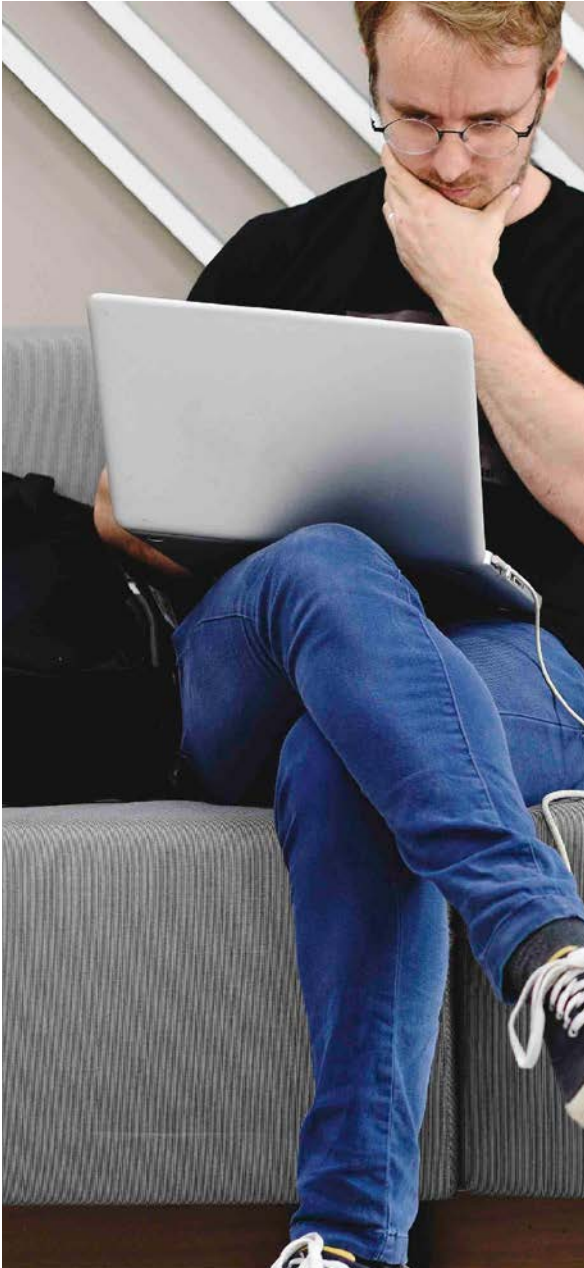
An Introduction to Root Cause Analysis

Describing the RCA process is like peeling back an onion. Every step of the process is itself comprised of several steps. These are the three main steps of the RCA process. The first two steps are often completed in tandem, either by an individual or by a team in a post-mortem incident review meeting.

-  **Correctly and completely identify the event or problem**
-  **Establish a timeline from normal operation through to the event**
-  **Separate root causes from causal factors**

01

Identifying the Event or Problem



Completely and accurately identifying the problem or event is perhaps the easiest part of RCA when it comes to networking issues.

That's right, I said easiest.

It's easy because all you have to do is ask yourself Why? When you have an answer to the question Why ask yourself why that thing occurred. Keep asking yourself Why until you can't ask it anymore – usually that's somewhere from 4-5 times. This process is often referred to as the 5 Whys.

Many engineers advocate utilizing an Ishikawa, or fishbone diagram to help organize the answers you collect to the 5 Whys. I like this myself, and often utilize a whiteboard and sticky notes while working the problem. If you prefer using a software diagramming tool that's fine, just use what is comfortable for you.

Example:

The Power of Why

Here's a real-world example FirstWave's Professional Services team encountered while conducting onsite training in system operation. An internal user called into the client's help desk and reported poor audio quality during a GoToMeeting with a potential customer.

Why? – A user reported poor voice quality during a GoToMeeting (first why)

Why? – Router interface that services switch to user's desktop experiencing high ifOutUtil (second why)

Why? – Server backups running during business hours (third why)

Why? – cron job running backup scripts set to run at 9 pm local timezone (fourth why)

Why? – Server running cron job is configured in UTC (fifth why)



The team started with the initial problem as reported and asked themselves Why is this happening. From there, they quickly came up with several spot checks and pulled performance data from the switch the user's desktop was connected to, and the upstream router for that switch; this identified a bandwidth bottleneck at the router and gave us our second Why.

Once the bandwidth bottleneck was identified, the engineers used opFlow, FirstWave's Netflow/IPFix analyzer to identify where the traffic through the router interface was originating from. This gave them the backup server, and a quick check of running tasks identified the backup job – and there the third Why was identified.

System backups were handled by a cron job, which was scheduled for 9 pm. A comment in the cron job suggested this was meant to be 9 pm local timezone (EST) to the server's physical location. This gave the team the fourth Why.

A check of the server's date and time indicated the server was configured for UTC, which gave us the fifth Why.

Not every problem analysis will be this simple, or straightforward. By organizing your Why questions, and their answers, into a fishbone diagram you will identify causes (and causal factors) leading to a problem definition and root cause. In short, keep asking Why until you can't ask it any further – this is usually where your problem starts.

02

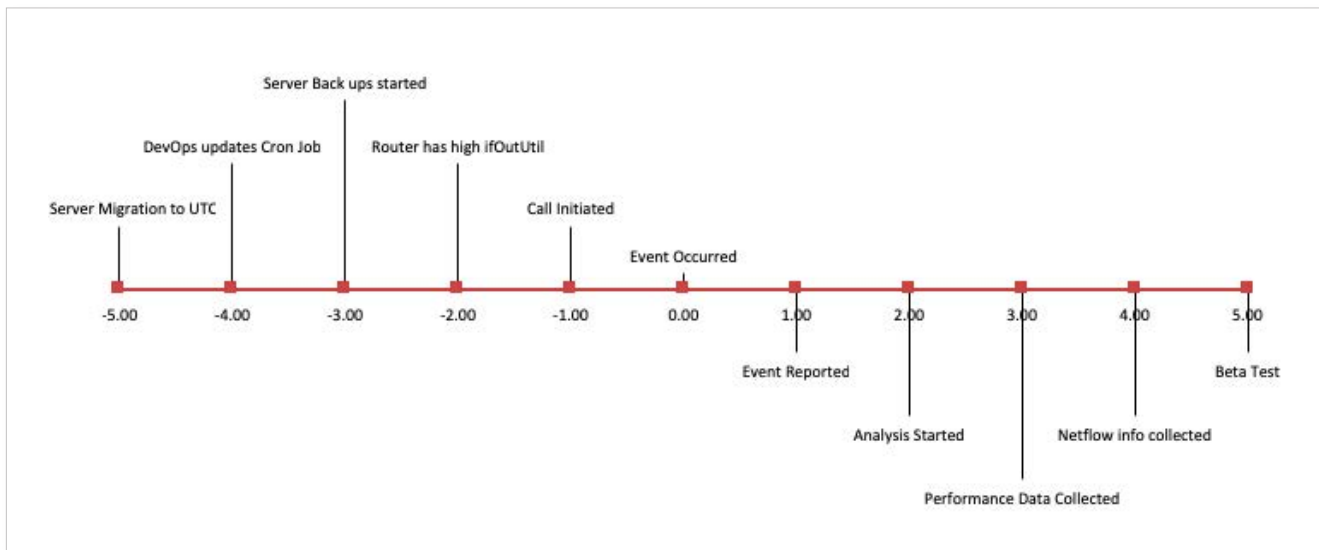
Establish a Timeline

When establishing a timeline it's important to investigate both the micro (this event's occurrence) and the macro (has this event occurred in the past). Thinking back to grade school mathematics, I draw a timeline, a horizontal line, across my whiteboard. In the center I place a dot – this is T0 (time zero) when the event occurred.

To the right of T0 I add tick marks for when additional details were discovered, when the user reported the issue, and when we collected performance or NetFlow information. I also add in marks for when other symptoms occurred or were reported, and for any additional NMS raised events.

To the left of the T0, I place everything we learned from asking Why – when did the backups start, when should they have started? I also review my NMS for events leading up to the performance issue; was interface utilization slowly on the rise, or did it jump dramatically?

Once I have mapped the micro timeline for this one occurrence I begin to look back through my data. This is where having a good depth of time-related performance information comes in handy. FirstWave's Network Management Information System (NMIS) can store detailed polling data indefinitely which allows fast visual analysis for time-based recurring events.



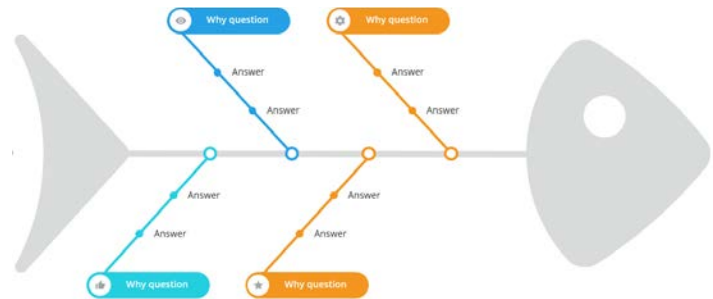
Example:

The Power of Time

As the engineers worked through their Why questions and built a fishbone diagram, they also created a timeline of the event.

They started defining T0 as when the event was reported, but as they collected data adjusted this to when the impact on the network actually started.

To the right of T0, they added in when the user reported the problem, when they started the event analysis, when performance data was collected from the switch and router, and the NetFlow information from the NetFlow collector. They also add in marks when other users reported performance impacts, and when NMIS raised events for rising ifOutUtil on both the router and backup server interfaces.



To the left of T0, they added when the backups started as well as when they should have started. They reviewed NMIS and found the initial minor, major, and warning events for rising iflOutUtil on the router interface.

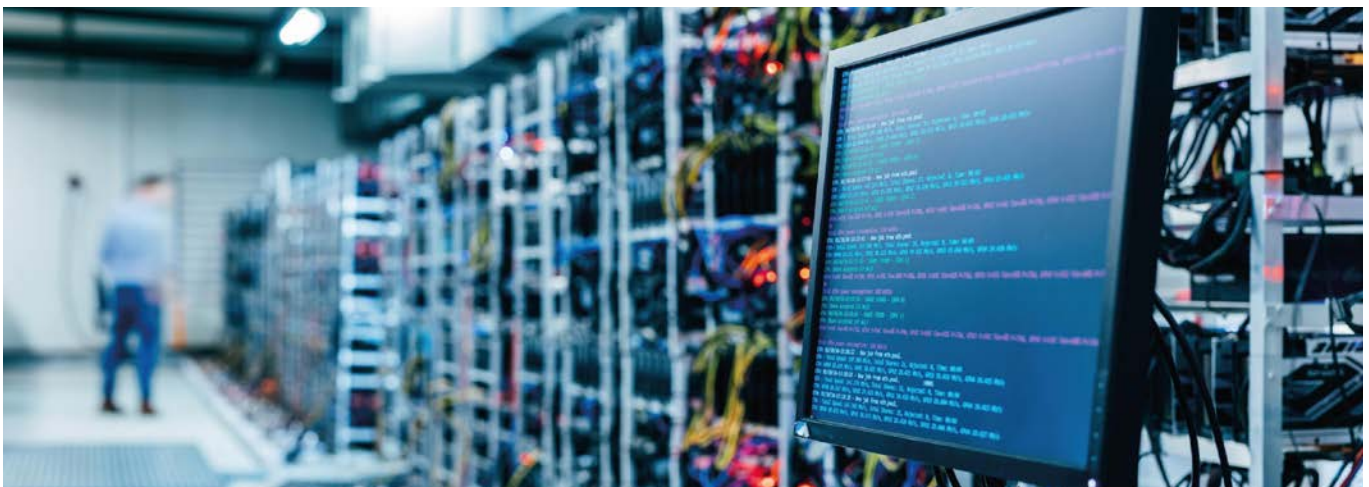
Once the timeline was complete the engineering team went on to look for past occurrences of this event. By widening the scale on the router's interface graphs the engineers could instantly see this interface had been reporting high ifOutUtil at the same time every weekday for several weeks. This cyclic behavior suggested it was a timed process and not a one-time user related event.

03

Root Causes vs. Causal Factors

As you build out and answer your Why questions you will inevitably discover several possible endpoints, each a potential root cause. However, some of these issues will simply be an issue caused by the root cause – a causal factor – and not the root cause itself.

It is critical to effecting long-term improvements in network performance that these causal factors be identified for what they are, and not misrepresented as the root cause.





Example:

Distracted by a Causal Factor

The engineering team quickly realized that all servers had, at one time on the past, been configured for local timezones and had only recently been standardized to UTC. While a process had been in place to identify schedules, like this cron job, and update them for the change to UTC, this one had been missed.

Some members of the team wanted to stop here and fix the cron schedule. However, the wider group asked: Why was a cron job for a critical process, on a critical server, missed in the update process?

Pulling the list of processes and files from the update team's records showed this file HAD been identified and updated, testing had been completed and verified. This brought about the next question: Why was the updated cron job changed, by who/what process?

While you can address causal factors they are often just a temporary fix or workaround for the larger issue. Sometimes this is all that can be done at the time, but if you don't identify and completely address the root cause any temporary solutions you apply to causal factors will break down over time.

Example:

Finding the Root Cause

Upon digging further, the engineers discovered that the cron job had been properly updated, but an older archived version of the cron job had been copied onto the server via a DevOps process. A Tiger Team was put together to research the DevOps archive and determine the extent of the problem. The Tiger Team reported back to the engineering group the next day; other outdated schedule files were found and also corrected.

The engineering team worked with the DevOps engineers to put a process in place to keep the DevOps file archive updated.



Closing out the Event

At this point, you've completed the Root Cause Analysis and identified the core issue causing the reported performance degradation. Since you've addressed the root cause this issue should not reoccur again. However, you can't stop here – there are two follow-up steps that are critical to your future success as an engineering organization

1. Document the issue

I like to use a centralized wiki, like Atlassian's Confluence, to capture my organization's knowledge in a central repository available for the entire engineering team. Here I will document the entire event, what was reported by the user, the performance information we captured, the RCA process and the end result – how and what we did to prevent this from happening again. Using tools like FirstWave's opEvents, I can then relate this wiki entry to the server, router, interfaces, and ifOutUtil event so if it reoccurs future engineers will have this reference available to them.

2. Follow-Up

The root cause has been identified, a remediated put in place, and a process developed to preclude it from happening again. However, this doesn't mean we're done with the troubleshooting. There are times where what we assume is the root cause is, in fact, just a causal factor. If that is the case, this problem will reassert itself again as the solution would only be a workaround for the true problem. The solution is to put a process in place, usually through a working group or engineering team meeting, to discuss user impacting events and determine if they relate to either open or remediated

What I've presented here is a simplified root cause analysis process. There are additional steps that can be taken depending on the type of equipment or process involved, but if you start here you can always grow the process as your team absorbs the process and methodology.



Mark Henry

Senior System Engineer

Mark Henry is a Senior System Engineer in FirstWave's North American office located in Charlotte, NC. A serial entrepreneur, Mark specializes in the ground-level, tactical application of real world processes including Agile Development, Test Driven Design, and Six-Sigma to deliver solutions that often outpace the market and his competition. Mark resides in Concord, NC with his wife Beth, their 8 children, and a dog named Jack Daniels.

FirstWave

If you'd like some guidance around setup and configuration, or If there's a specific issue you're trying to solve in your network environment - click below to request a demo from an FirstWave engineer.

[Book a demo](#)

**Thank
You.**

FirstWave

Our passion is to create intelligent software that our service provider partners and customers love.

Get Expert
Solutions

Book a demo

FirstWave is a publicly-listed, global technology company formed in 2004 in Sydney, Australia. FirstWave's globally unique CyberCision™ platform provides best-in-class cybersecurity technologies, enabling FirstWave's Partners, including some of the world's largest telcos and managed service providers (MSPs), to protect their customers from cyber-attacks, while rapidly growing cybersecurity services revenues at scale.

In January 2022, FirstWave acquired FirstWave Limited (FirstWave), a leading provider of enterprise-grade network management, automation and IT audit software, with 150,000 organisations using their software across 178 countries and enterprise clients including Microsoft, Telmex, Claro, NextLink and NASA.

Integrating CyberCision™ with FirstWave's flagship Network Management Information System (NMIS) and Open-Audit product enables FirstWave to provide a comprehensive end-to-end solution for network discovery, management and cybersecurity for its Partners globally.

With over 150,000 organisations now using FirstWave technology, we are well positioned to be a leader of transformational change in the IT Operations and Cybersecurity world.