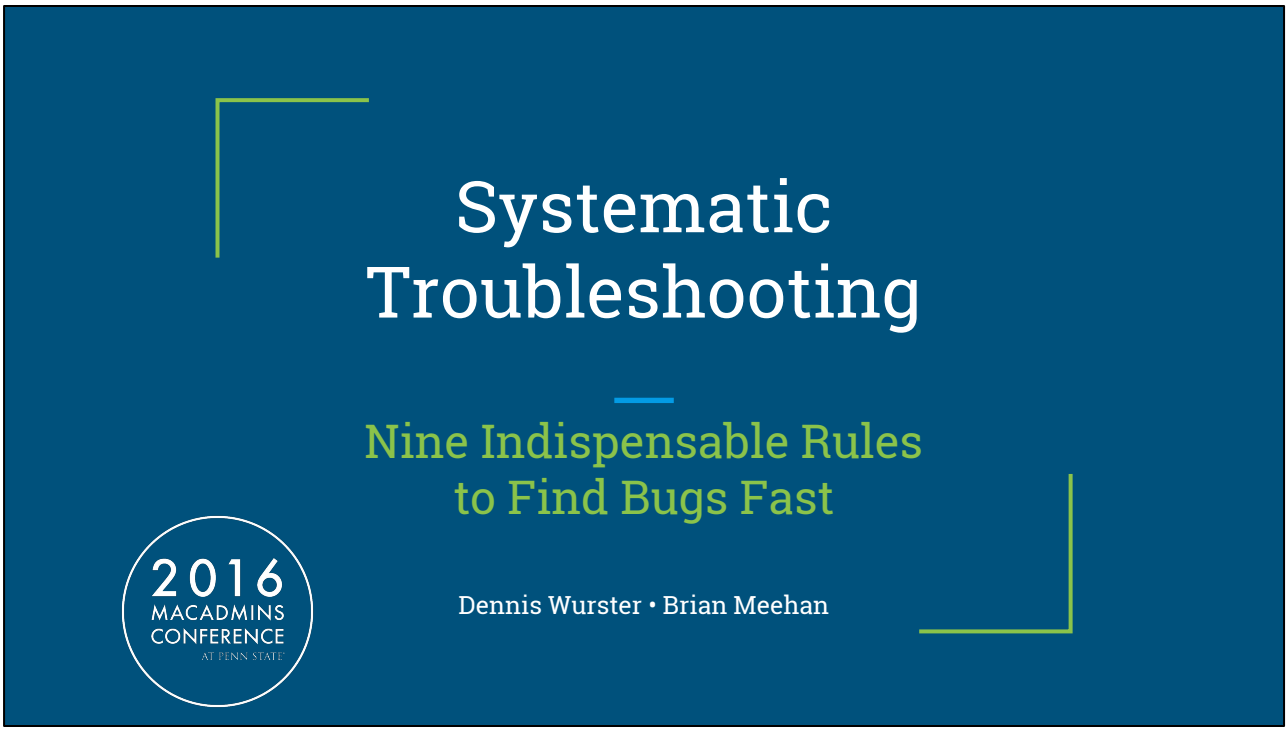
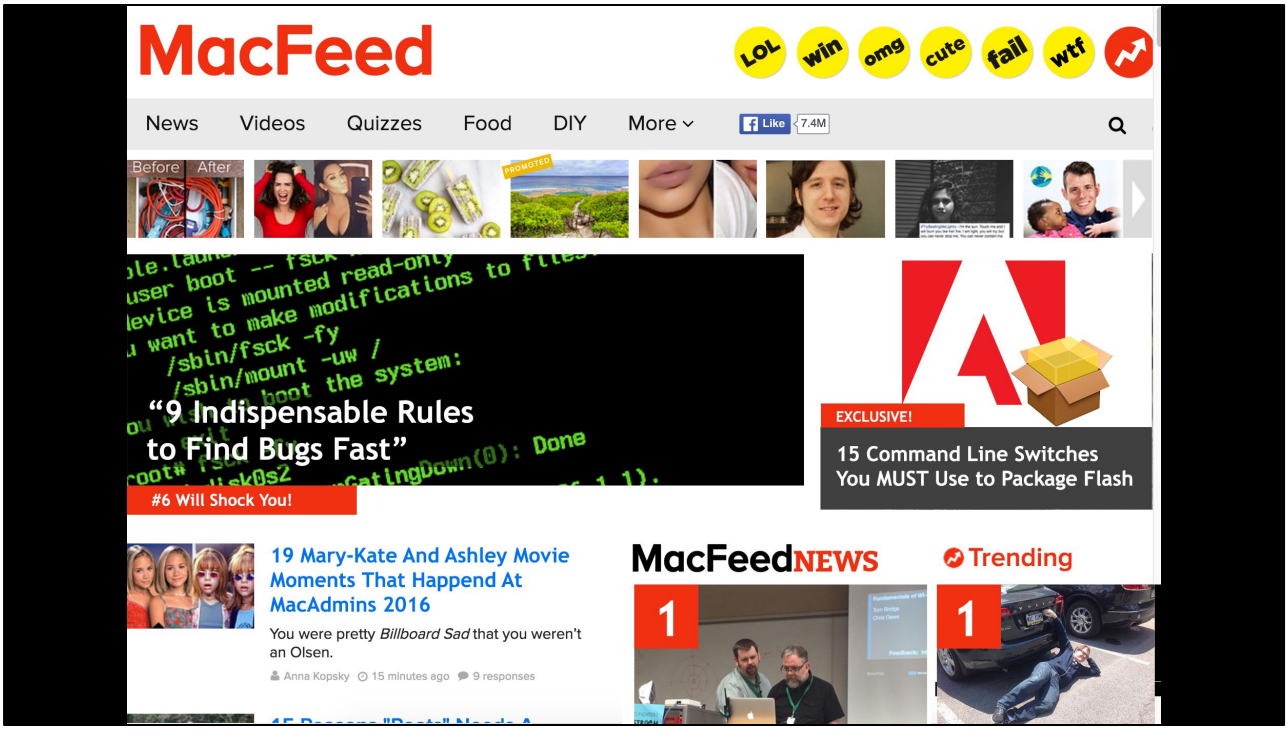


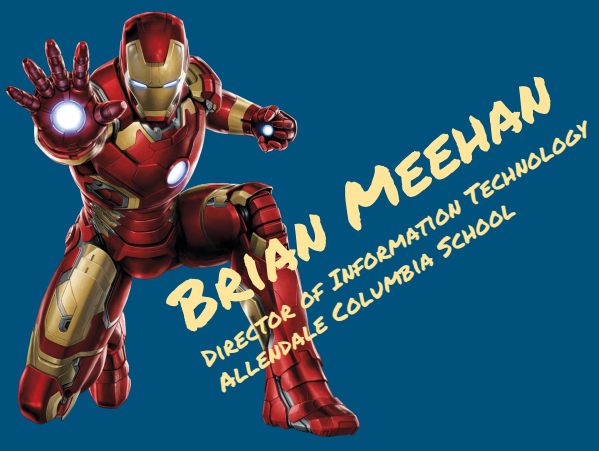
Ladies and Gentlemen, welcome to Systematic troubleshooting. Dennis and I are glad you're here. The next 75 minutes may very well change your life. At the very least, you'll be… at least an hour older.



Okay, Okay, we know what you’re thinking. The title is a little click-bait-ey.

But that’s the point. You gotta have a schtick when you’re up right after breakfast and stacked against PSU in one room and Greg Neagle in another.

These guys...



Please allow us to introduce ourselves.

I'm Dennis…
(identify)

And I'm Brian…
(identify)

You might remember us from such sessions as, PSUMAC 2014's "Sysadmin 101"

This Is Obvious

Can you believe that the PSUMAC conference committee greenlit this topic? Who could possibly fill 75 minutes about the most basic of our skills, which we clearly all already possess?

There are rules to troubleshooting. Guidelines. And because you're all here to learn them tells us two things.

1) The things are here are obvious, and you probably know them already, but you don't KNOW you know them, or you've never seen them projected in 4000 lumens in 16" letters
2) You decided to come here and find out how to be better at troubleshooting, so it must be important

The rules that we're going to teach you today ARE obvious, and they're not that hard. But the tricky part is in remembering them when the time counts, and how to apply them.

Troubleshooting is a universal skill. You can apply these rules to all kinds of problems that you run into. I've also worked as an EMT, and I would approach my patients with the same systematic methods that I used as an IT professional.

OK, so not all problems. The economy is too complex. When your wife asks you which dress she should wear… there is no right answer.

Debugging != Troubleshooting

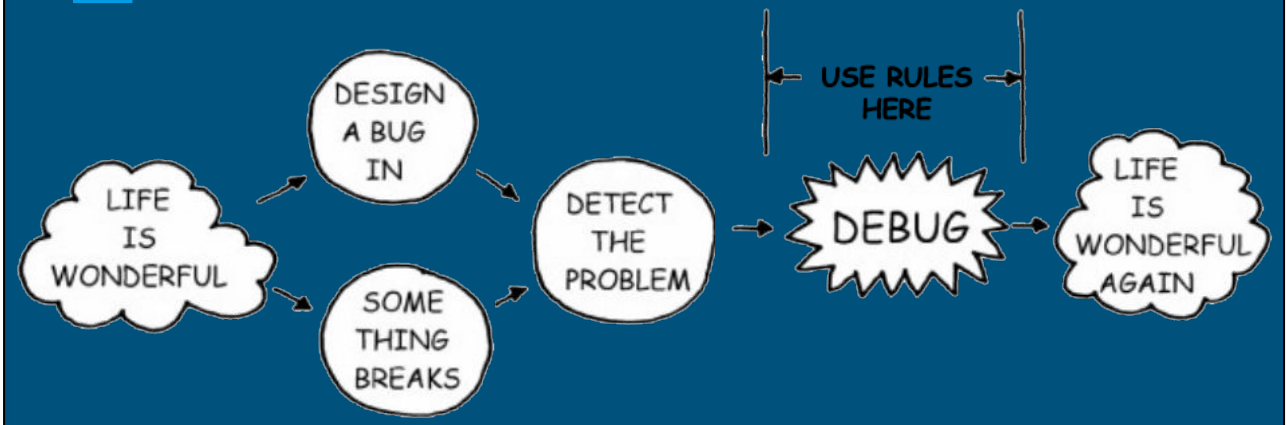
So the talk is called “Troubleshooting”, but (as you’ll see in a minute) it’s based on a book titled “Debugging”? What’s the deal?

This is *mostly* a semantic difference, but for the purposes of this session, debugging happens at design time, and troubleshooting happens after release. We’re in a unique position as Mac Admins in that we find ourselves sometimes in both groups. Debugging finds a problem with a design, and troubleshooting finds what’s broken with a otherwise known-good design.

For example - the team working on designing the 2020 Volvo S90 will debug. But finding out what might be wrong with Mike Boylan’s volvo… that needs troubleshooting.

Doctors troubleshoot. They weren’t invited to the design meeting.

When will I ever use this?



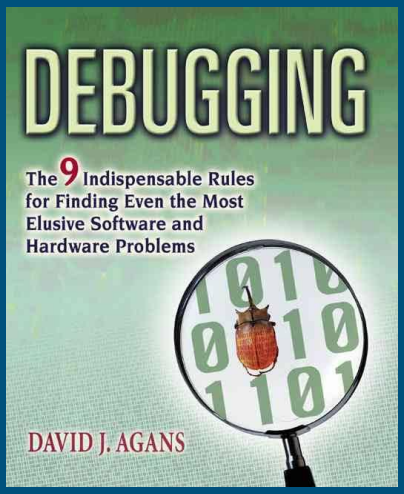
So we've made the case that this is important, but where does it fit into my life as a Sysadmin?

Click We start our days in blissful ignorance, where all is well.
Click And then something isn't. Either during the design phase where a bug is introduced, or a ticket is submitted for something broken in production.
Click And there it is.
Click So we put on our cape and tights and apply the principles of systematic troubleshooting
Click and then all is well with the world again. Go back to reading Reddit.

And now...

The rules!

Each rule will be presented with :
WHAT - A general description
WHY - The value derived
HOW - Ideas and suggestions for implementation of for the modern MacAdmin



The Nine Indispensable Rules

1. Understand the system
2. Make it fail
3. Quit thinking and look
4. Divide and Conquer
5. Change one thing at a time
6. Keep an audit trail
7. Check the plug
8. Get a fresh view
9. If you didn't fix it, it ain't fixed.

These rules come from a book that I read several years ago written by a man named David Agans. He is an electrical engineer and the book is full of examples of how he used these rules to solve problems. You should check it out. It's less than 200 pages long, so you won't spend a lot of time reading it.

There they are! So buckle up, 'cause here we go!

What

a. The magic phrase "Theory of Operation"
   i. A.K.A. "So here's what we had in mind when we made this…"
b. Read the Manual
c. No really, Read the Manual -- thoroughly! **Click for dremel. Click again for emphasis after explanation.**

Why

1. Don't trust your memory! It tricks you into thinking that it's right, even when it's not.
2. Read the Manual BEFORE something fails

How

d. Keep manuals at-the-ready, in a repository
e. Scan paper manuals if a PDF version isn't available
   i. Photo needed: ScanSnap, HP Scanner; Xerox WorkCentre
   ii. Enable OCR
f. Good Sources: Peachpit OSX Training books, Apple Dev Account; Google searches with "type:pdf" and specific model numbers.
g. Don't discount the power of Benchmarking.
   i. "My computer is slow"
   ii. "Chainsaws are supposed to be loud"

Story

Troubleshooting email involves a lot of working parts. A technique that i like to use sometimes is manually sending an email via telnet. You can open a telnet session from the MacOS terminal on port 25 and communicate directly, command-by-command, with the email server. Knowing the correct commands and proper order is essential. It's documented in section 3 of IETF RFC 2821. Different mail servers give different responses and it's important to know what to expect back as a response to know if your system is working correctly.

What

- Some failures are HARD FAILURES, some are PERIODIC FAILURES, some are (seemingly) RANDOM

Why

- So you can look at it. One failure is never enough
- So you can focus on the cause
- So you can tell if you fixed it. Having a sure-fire way to make it fail builds your success test.
    - Did you fix it, or did you just get lucky, punk?

How

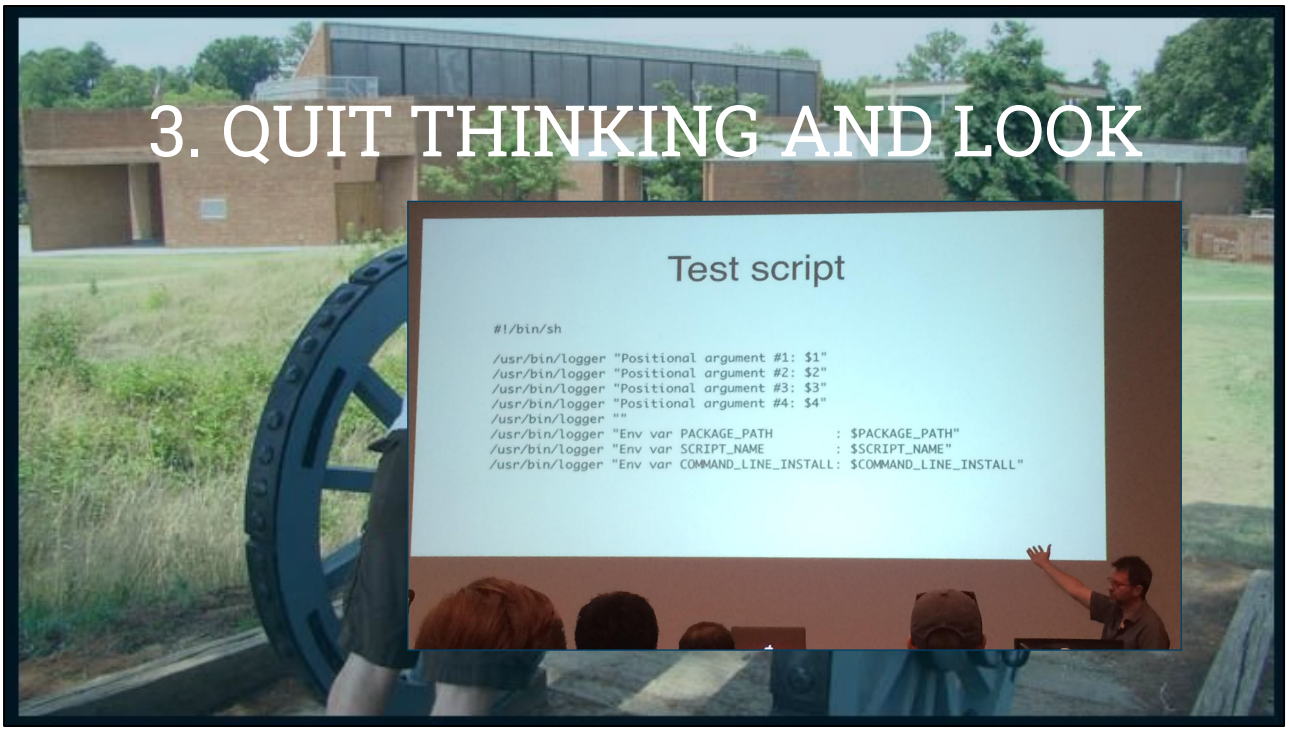
- Make it fail multiple times - write down your procedure and do it again
- Start at the beginning
- STIMULATE the fault, but don't SIMulate the fault
- Sometimes it's intermittent. This is true. That's why we have logs and other types of output, and bulk text editors that can look for patterns. And if it's really, really intermittent, consider putting a known pattern of data through your system.

Story

War Story. My house had a window that leaked only when it rained hard and the wind was from the southeast. I didn't wait for the next big storm; I got out there with a ladder and a hose and made it leak. This allowed me to see exactly where the water

was getting in, determine that there was a gap in the caulking, and, after I caulked it, verify that even under hose pressure the leak was fixed.

(skip) War Story. WiFi authentication at Allendale Columbia. Monitor at AD Radius Server, Wireshark on the AP, Wireless diagnostics on the end point with tail logs open. Now turn off and turn on wifi… again and again...



As Sysadmins, we love to provide answers. Often, that means we jump to conclusions, which can cloud our objectivity.

What

- See the failure (first-hand)!
- See the details
  - For example, you walk into a room. You flick the light switch. It doesn't come on. The fact that it's still dark is not the problem - it's just one symptom.
  - Hey Brian: How many sysadmins does it take to change a lightbulb?
    Dunno. I can't reproduce the problem. The bulb in the IT office seems to be working fine.
    Close the ticket.
  - We all have the users that call us and tell us, "The printer isn't working" That may be all of the information that you get. You need to go and see for yourself.

Why

- By simply guessing or going on partial descriptions, you end up fixing what you guessed, but it was something completely different that failed.
- "I need the admin password" - Is it something that legitimately needs approval? Is it something future you could automate? Or did some helpful website suggest that your user needed an upgrade to the Flash Player Pro?

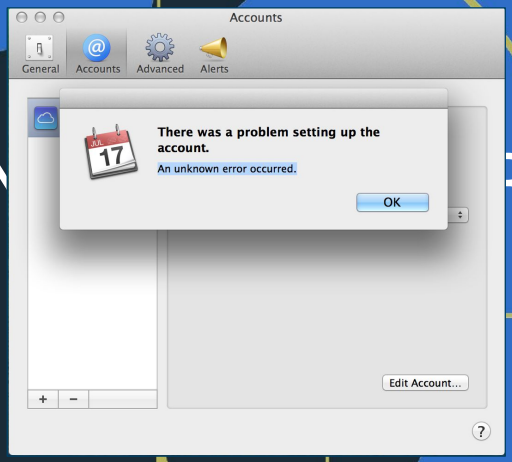
How

- Guess, but only to focus the search.
- Try the easy/fast fix first. You'll still want to document what change you made to cause the problem to stop happening.
- Run with DEBUG MODE or 'verbose mode' on. Check your tools for more instructions on how to do this.
- **Click** Instrument the system, as Greg has done here in this screenshot from yesterday's session. Here he's talking about how he checks the values of variables in his install scripts
- If you're watching a log file on a remote computer via SSH, use the 'tail -f' command
- Teach users to take screen shots, **and thank them when they do**

Story

Windows 10 upgrade at Midland Appraisal, old video card, RDP

4. DIVIDE AN R



What

- “Successive Approximation” / ”Split-half”/ “Binary” searches
- Have you ever played Battleship as a kid? You prepare to battle against your opponent. You aim your cannons. A6 FIRE! Not even close. Adjust, A10 FIRE! Nope. Split the difference… A8 Direct hit!
  Of course, it’s more important to troubleshoot quickly in a situation where trouble is shooting at you.

Why

- Get ‘in the ballpark’ to define a range. Demo “Pick a number 1-100”. 7 guesses.

How

- Some workflows are really long, and take a long time to execute. Start with the bad. Think of a river that’s being polluted. Start where the river smells bad and work upstream. Start where the problem is.
- **Click** Sometimes error messages are too vague. This is where knowing the system works to your advantage
- We recommend Injecting easy-to-spot patterns, like phrases you can search for in a log file or other output.

Story

I had a professor that was trying to send out a group email. He sent it out, but got a message back saying, “I’m sorry, but this is undeliverable”. I was really busy, and the response he got back was pretty vague. I asked him to cut out

half the addresses and try again. After multiple successive reductions, we found that an email address in his contact book was missing the top level domain, “.com”. Not only did he fix this problem, but he corrected the address book entry to prevent future problems.

5. CHANGE ONE THING AT A TIME

What

- The 5th indispensable rule for finding bugs fast is “Changing only one thing at a time”

Why

- When you make multiple adjustments at once, that approach may solve the problem, or it may cause more problems. It won’t help you prevent it in the future, because you won’t be able to say for certain which was the real cause.
- Worse, all your changes can break something else that was fine to begin with.

How

- As we mentioned earlier today many of these steps are going to seem very very obvious and this one is the one that I found to be the most obvious.
- Changing one thing at a time is probably one of the most important things you can do because it keeps you from making extra work for yourself. It is important to go into this process by having a plan; figuring out precisely what you're going to do first, how you're going to document it, and how you’re going to record the results.
- Then just as a rule states, you’re going to change one thing and only one thing and see if the results change. If they don't change, move on & change the first thing back. Now move to the next variable. Follow your plan. Compare your results to a product or system that works as it's supposed to. One of the questions you can ask is, “When's the last time this work the way it was

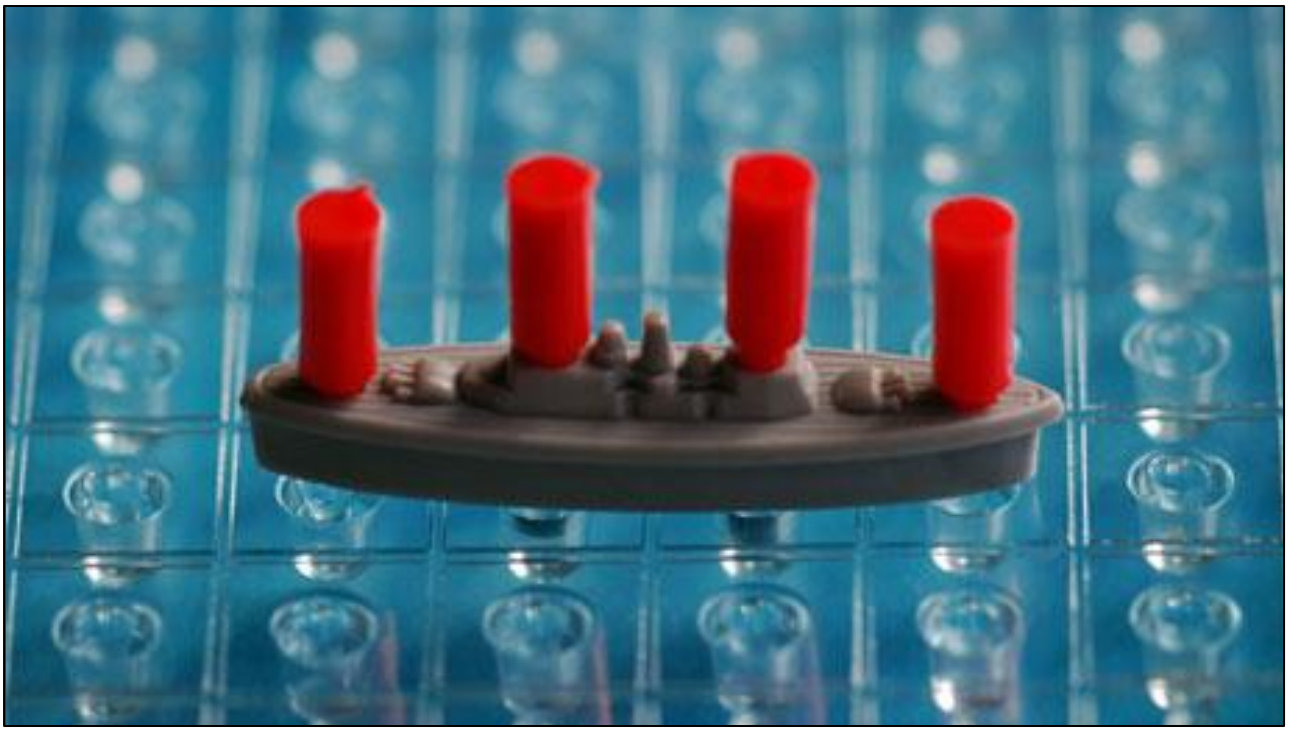
- supposed to?” That helps you get a sense of what may have changed in the meantime.

Story

Where I work, we’ve recently undergone a printer transition. When the vendor brought in one of the new printers, it could not connect to the network. He started suggesting multiple solutions, but we took a systematic approach of changing just one thing at a time.

- First, we swapped printers and found the old one worked, and the new one didn’t.
- Next we swapped cables, and found that the old one still worked with the different cable, but the new one didn’t.
- Next, we replaced the network jack, and then the new printer worked.

This helped us figure out the root cause - poor termination made the old printer run at a slower speed. Once the network jack was replaced, the new printer worked fine. We got there quickly by changing just one thing at a time.



What

- Rule #6… Keep an audit trail. Leave yourself some breadcrumbs.
- Write it ALL down, As soon as possible, in as much detail as you can stand!
- Write down what you did, in what order, and what happened next (your results)

Why

- The devil is in the details. Know the ACTUAL symptoms and when they happen, NOT just a submitted logfile.
- There are two types of symptoms: Subjective and Objective. In other words, “what you’ve been told and what you’ve see for yourself”.
- Don’t assume your memory is perfect. It’s not
- Don’t assume you are the only person that has or ever will work on this problem. Accurate documentation will help you for tasks that you only do infrequently, or that you want to offload to someone else.

How

-
- MacOS’s Console utility gives you access to important system logs. Open multiple log files at once by keeping each one it its own window.
- If you have a really long log file, you might consider a specialty text editor like TextWrangler or BBEdit. They’ve got great searching abilities for really large files, and they’ve got a feature that allows you to scroll two side-by-side

- windows at the same time
- Lastly, if you are working across multiple log files, make sure you've enabled time stamps in your log files and you use them to connect events.

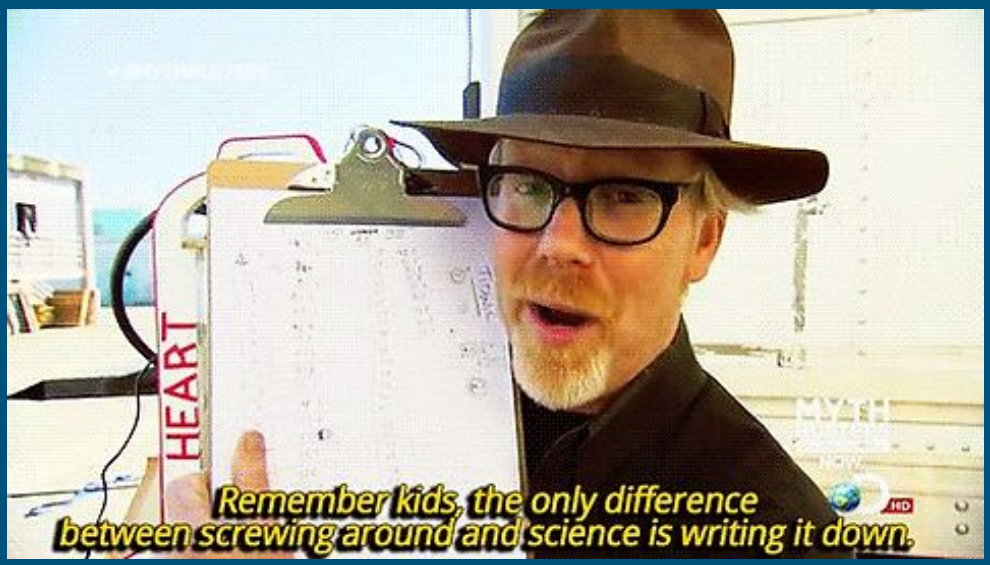
Story

This story is a little long, but it's a great example. Last November, just before grade reports were due at our school, teachers let me know en masse that we were having a problem with our Student Information System. When they submitted any form, the pages that normally responded in seconds were taking between 4 and 6 minutes to submit, if at all. "The wi-fi is slow" was their chant, but I quickly ruled that out with solid benchmarks across all other sites. The heat was on with report cards looming, so I called the vendor of our SIS, which is just 2 developers who live in Connecticut and British Columbia, respectively. Of course, neither of them noticed any problem. They even screen shared with me to prove it. Was there really something wrong on our network side? Data to the rescue!

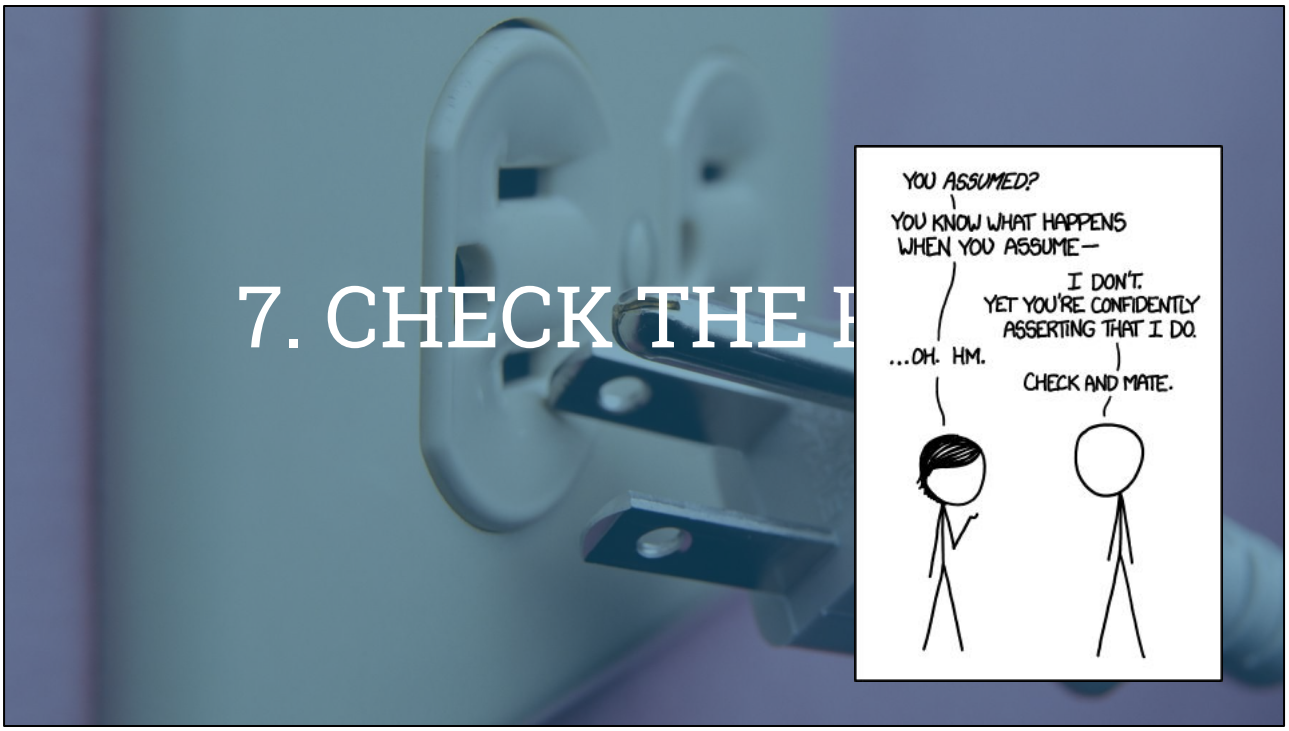
I started looking at firewall logs between our network and their server, and found a very high rate of retransmits and time outs. Of course, that made sense, but why?

If it wasn't their server, and it wasn't our network, then it must be something in between. I created a document to record tracert results from multiple locations… our school, my house, various faculty member's homes, and a few side-clients where I could test. They spanned the gamut of connectivity types in our city. Our vendor participated in this exercise too. When I looked at the results, it quickly became apparent that this was a Rochester problem. BC and CT were fine, but the DSL and cable modem users in Rochester all fell prey to a tracert that didn't complete after a hop coming off Cogent and onto a network called "UnifiedLayer". I called up Cogent and got a person on the 2nd try! She was very helpful and confirmed that the IP addresses I gave her for UnifiedLayer belonged to a Cogent client. Then she gave me the name and phone number to contact them. That turned out to be a direct desk number for the network manager of the Co-lo facility where the hosting provider serving our website operates. I shared my documents with him and even found out that I could experience network latency to multiple web sites in the same IP range as our SIS host. Then next day, he called back and thanked me for helping them identify a failed port on the switch connected to the cabinet where our hosting server was. They replaced the switch overnight, and our performance was back to normal.

Click - Battleship incorporates the elements of "Divide and Conquer", "Change One Thing At a Time" and "Keep an Audit Trail" <click> for Adam Savage.



It’s just like they say on Mythbusters...



What

- Sometimes it's really that simple. Don't make assumption. Live in an observation & conclusions, based on real evidence.

Why

- **Click.** Assumptions are often faulty.
- They can lead you down the path of doing more work. I don't know about you, but I don't like to do any more work than I have to.

How

- So don't start at square-three. Make sure your start up conditions are the same every time you test.
- If you were troubleshooting the installation of a package via munki, you might head to the command line and run managedsoftwareupdate -vvvv. First you'd see if it could resolve your munki server's IP. Then, can it connect via HTTP. Then does it find a manifest. Then does it have a valid catalog defined… etc cetera.
- When we approach networking problems, we use the familiar OSI model of
- Test the tool (Dead batteries in the continuity-meter make the measurements useless)
- Consider the use of virtual machines and snapshots so you can try work flows from the exact same starting conditions.

Story

This is one I overheard at PSUMac yesterday. An attendee was telling about a traffic

accident outside of his house, where a tall truck ripped the power and utility wires down and off the side of his house. We're talking fire department, utility company and the like. He called his Internet provider to let them know, and prefaced with, "Before you tell me to reboot my modem, I need to tell you that the wires were ripped off the side of my house by a truck". The support rep responded with, "OK, well let me check. Hmm… I can't see your modem. Let me dispatch a technician." Now a residential technician also isn't equipped to handle restringing cable across 5 lines of highway, but (Step 3, quick thinking and look) he was able to call his buddies and within 1 hour, bucket trucks were onsite and getting him back online.

What

- It can happen to all of us. We can get fatigue from troubleshooting the same problem for too long. We may run into something we've never seen before. It helps to have another set of eyes on the problem.

Why

- Don't spin your wheels when you don't have to. There's a cost:benefit to how long and how much effort you put into troubleshooting.

How

- Ask for fresh insights. Phone a friend.
- Tap expertise. Sometimes explaining the problem to someone else will clarify it for you.
- Listen to the voice of experience. There are people that have done this before you. YOU ARE NOT THE FIRST
- Know that help is all around you, like vendors and co-workers
- Don't be proud. You aren't expect to know all things at all times. This is 2016. Google is your friend.
- When you DO talk to others, report symptoms, NOT theories. Don't cloud your expert's view.
- Realize that you don't have to be sure

Story

Are you on Slack? <show of hands> What's Slack? It's the reason I owe Rich Trouton

an entire of Diet Coke! It’s the mac-admin hive-mind and the best real-time resource you could ask for.

What

- And now, the final tip. Did you fix it?

Why

- There's nothing more embarrassing than closing a ticket and then having to reopen it for the same problem. "It still doesn't work"
- You have to be sure that it was YOUR fix that fixed it. "I changed my wallpaper, and it stopped raining". Correlation does not equal causation.
- Nothing goes away by itself. Ever. It will rear its ugly head again.

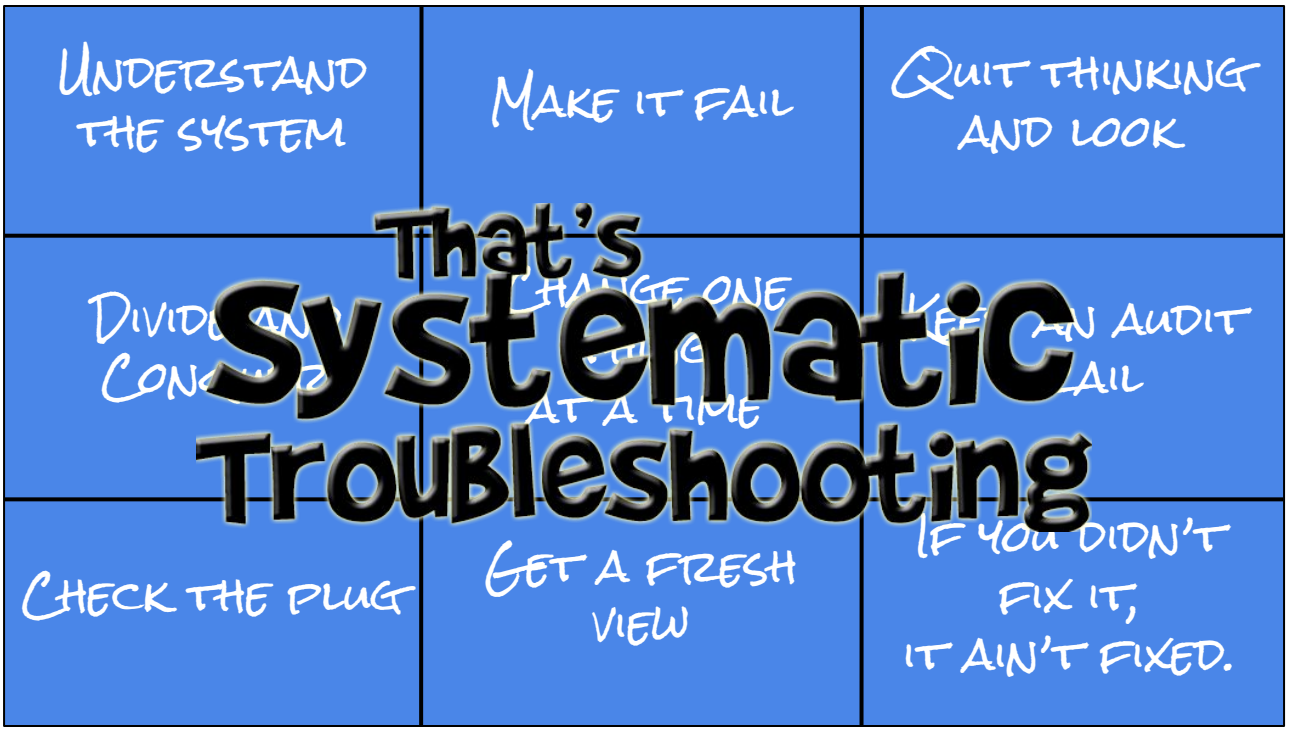
How

- Previously, we documented the steps it takes to cause the problem. So be consistent.
- Take whatever stimulated the failure and use those same steps to prove that the problem is now gone.
- Document the process used to fix the problem. In doing so, you might be able to prevent the problem in the future by improving processes, software or hardware.
- Install a trap in order to catch or log future instances of this happening again. This is a good time to update your monitoring events.
- This is the cleanup step.

Story

A while back, I was running into problems with a specific printer and 10.11. (Yes, I

know this is my 2nd printer story in the last half hour. Printers, amirite?)
The symptom was, “Um, It’s not printing”, but I had to **Quit thinking and Look** and realize that the jobs were getting stuck at the Fiery RIP. This ancient RIP wasn’t working with any vendor-provided drivers. So I **got a fresh view** by escalating to the printing vendor and supplied them with documentation I created while **keeping an audit trail**, including screenshots. The vendor chose to replace the RIP software, which allowed us to use newly-supplied drivers. Once installed, it was time to **make it fail** with some jobs that had caused us problems in the past. Once validated, we pushed out new drivers via Munki. It’s been running smoothly for weeks.



And those, ladies and gentleman, are the 9 indispensable tips for finding bugs fast!

Q & A

Let’s see how fast I can throw this thing.

Dennis Wurster - @wildeep

Brian Meehan - @binarydaze

Feedback: http://j.mp/psumac2016-41