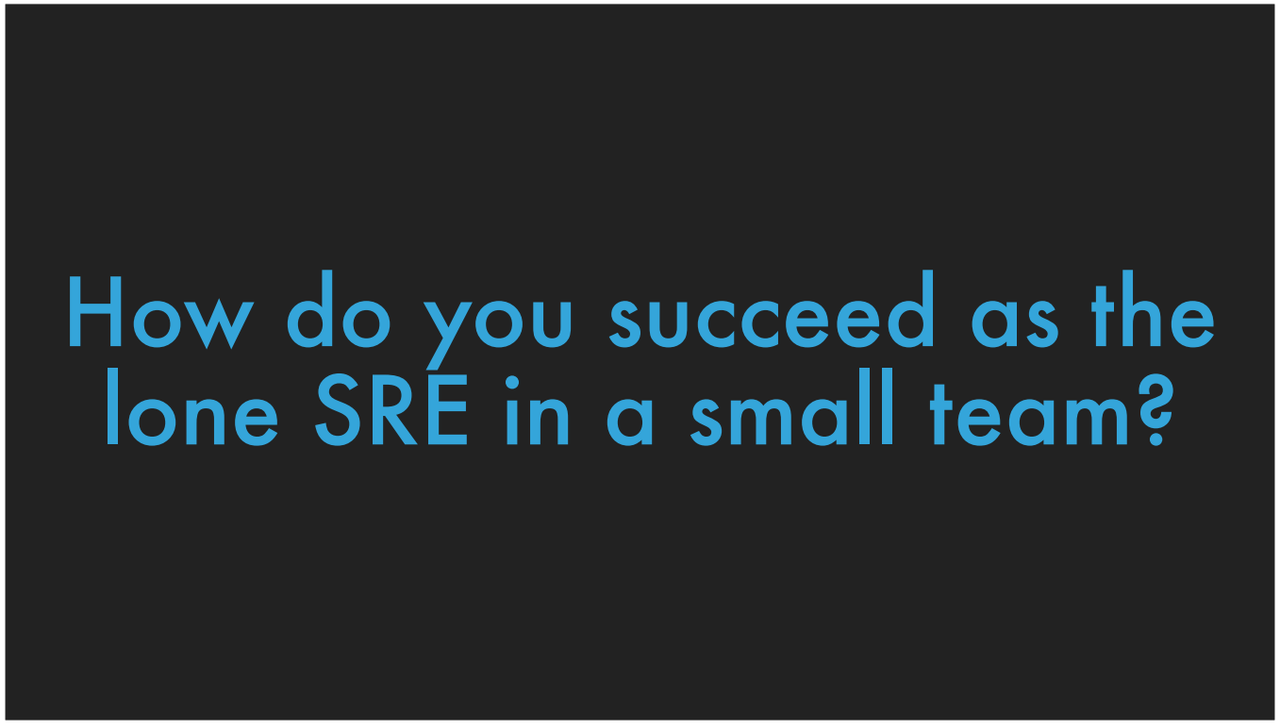
Danny Kirchmeier
Infrastructure Security Engineer @ Outschool
danny@kirchmeier.us
danny@outschool.com

# Succeeding as the Lone SRE in a Small Team




- “How do you succeed as the lone SRE in a small team?”
- I’ll be answering this question today based on my own learnings from working in small teams over the last 14 years.
- Let’s dissect this question before we get started to make sure we’re on the same page.

# How do you succeed as the lone SRE in a small team?

- “Small team”
- My career has been spent in the land of startups with a total engineer count ranging from 5 to 50.

# How do you succeed as the lone SRE in a small team?

- At the small team size, sometimes you'll have multiple people who can share the SRE burden, but there's often at least one person, maybe you, who gets to be the Atlas of the team, holding everything on your shoulders.
- This can be fun and exciting, yet it can also come with unique challenges.

# How do you succeed as the lone SRE in a small team?

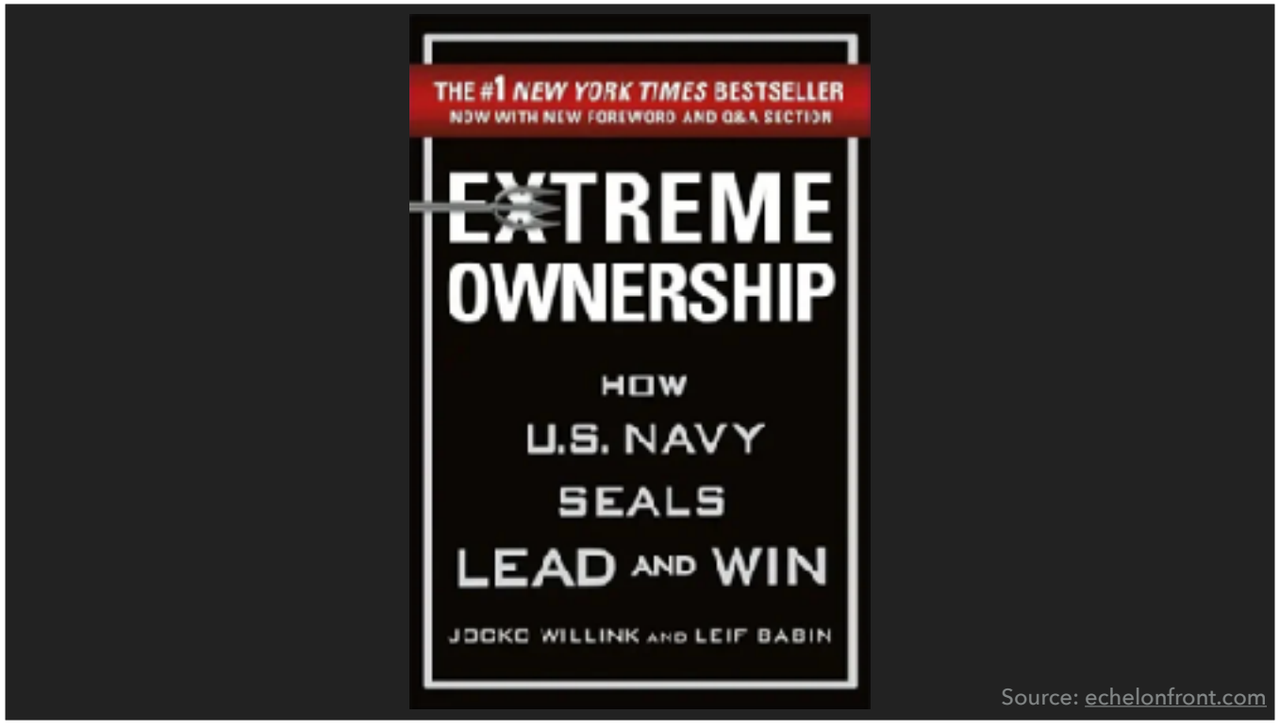
- But even in a large team, you still want to succeed right? But what is success?
- No, actually, a 15 minute talk isn't enough time to try and address that at a philosophical level, so I'm going to assume that you have an idea of what success looks like.
- But be it more money, business acclaim, professional advancement, personal growth or even improved relationships with others, I think that the one core component of this is:

# Ownership

- Ownership.
- But that's not anything special. Being responsible, completing your tasks, doing and identifying these things for yourself isn't anything new. So I want to take this and go more extreme and suggest that success can come through:

# Extreme Ownership

- Extreme Ownership.
- This isn't my idea, nor is it my own term. I've lived this out in principle longer than the book has been around,



Source: echelonfront.com

- but the phrase became popular by the 2015 book "Extreme Ownership: How U.S. Navy Seals Lead and Win".
- Brief aside: I enjoyed reading the book, but let me warn you, it is a very intense book. The authors provide American war stories along with business stories, but they put most of their focus into relating this to leadership examples, some that can be applied to technical leading, but others that aren't so relatable. But before I can talk through that, we need to know what extreme ownership means.
- So according to the book:

# Extreme Ownership says

You are responsible for not just those tasks which you directly control, but for all those that affect whether or not your mission is successful.

- Extreme Ownership Says
- “You are responsible for not just those tasks which you directly control, but for all those that affect whether or not your mission is successful.”
- So the key part of this is to think more holistically and about your mission. Like, imagine your friend asks you to bring a dessert to a party in a few days. People will enjoy those famous chocolate chip cookies of yours. But think bigger: Take it a step further and ask if there's a theme for the party? Then, figure out how "your mission" succeeds. If your mission to grow in other areas of your line then just show up for dinner. The cookies will be fine.
- Or Is your mission to impress your friends? In that case, it turns out that your friend is making their Sicilian grandmother's famous pasta so you decide making Tiramisu would be worthwhile.

# What is your mission as an SRE?

To keep the system reliable.

- And so, What is your mission as an SRE? As a system reliability engineer? You don't have to go complicated here. It's
- ***
- to keep the system reliable.
- Right? At a core, this is what we do. We don't typically talk our mission though, we talk about the direct effects we have on our systems.

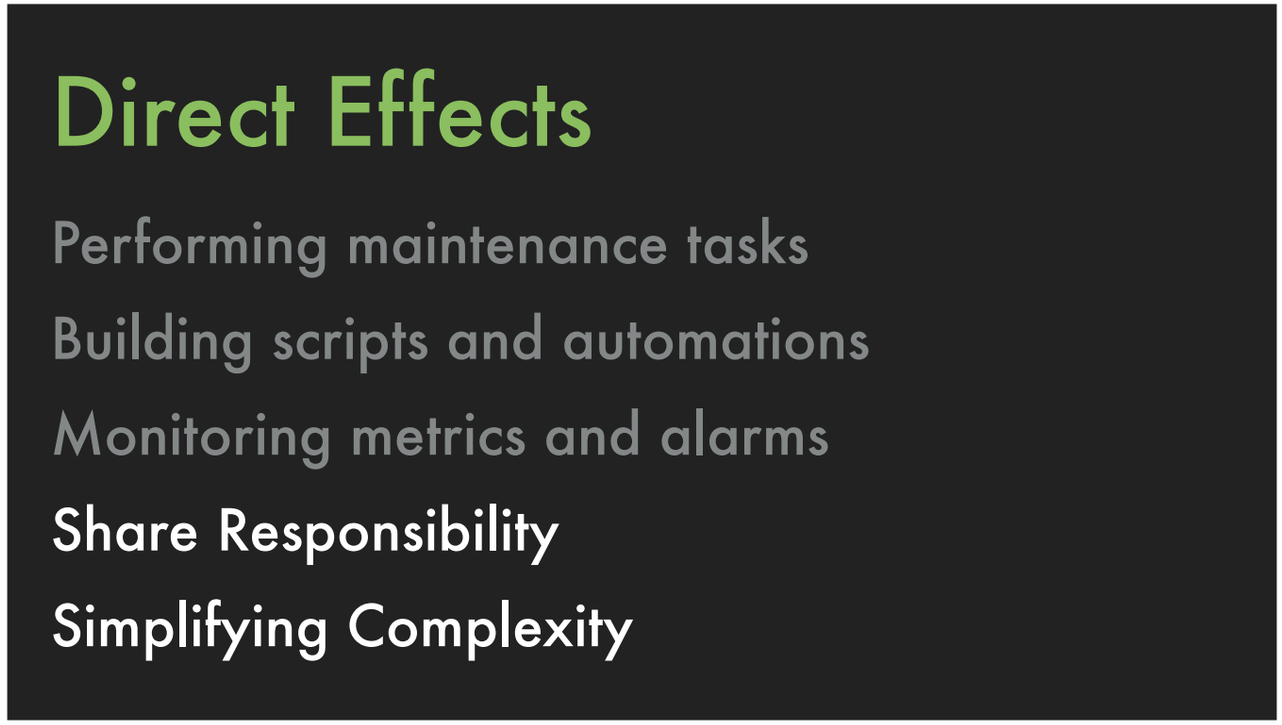
## Direct Effects

Performing maintenance tasks

Building scripts and automations

Monitoring metrics and alarms

- We'll talk about maintenance tasks or building scrips and automations or monitoring metrics and alarms.
- But I haven't said anything new, have I? We all take regular ownership of these tasks. And taking extreme ownership of these, well, that looks like what the senior and staff level SREs do. These sort of tasks are so ingrained into our discussions and learnings that we already recognize ways to own and improve them.
- There are two more direct effects SREs can have that are especially important for small teams:
-




- Sharing Responsibility and Simplifying Complexity.
- When you are one of the few SREs, being able to share your responsibilities with others is critically important. This will maximize your time as it allows others to operate with virtual versions of you through what you've shared. But you can't expect everyone else around you to know the system at the same depth as you, and so this is where you'll need to simplify, either by standardizing the edge-cases, or creating abstractions or interfaces that can hide away some of the intricacies of systems. You may even need to replace custom solutions with more standard tooling.
- And these two ideas require a great deal of initiative on your part, or you could say an extreme level of ownership, because systems naturally grow in complexity and people naturally don't want responsibility for things they don't understand. You will have to work continuously and with great creativity to fight those natural processes.
- But like I said, there's still plenty written about these direct effects we have. So I want to shift our focus away from them and over to
-

# Reactive Effects
# "How do you react?"

- Reactive Effects, or basically asking "How do you react?"
- After a scrum sprint or after a costly issue was fixed, does your team hold a retrospective meeting to look at what went right or wrong? If so, and if done correctly, you are already practicing a team wide level of ownership focused on reaction and how you can improve for the future.
-

# How do you react?

- But I want to shift the focus back on ourselves, on the YOU specifically. Because this the core of extreme ownership, not just the team taking ownership of issues but you specifically as a singular being. Even if you're not the lone SRE, even among a team, you still need to take responsibility for your own actions and this is what I want to ask. How do you react...
-

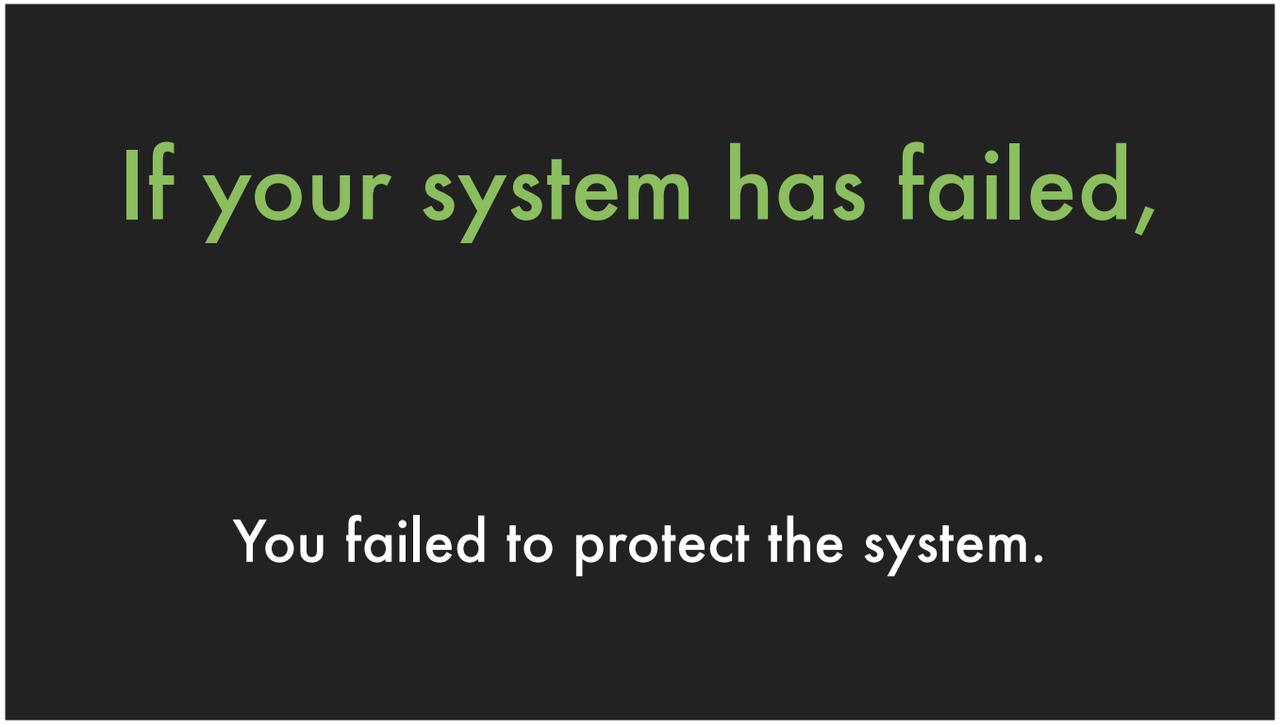
## How do you react when...

your system has failed?

someone has questions about your system?

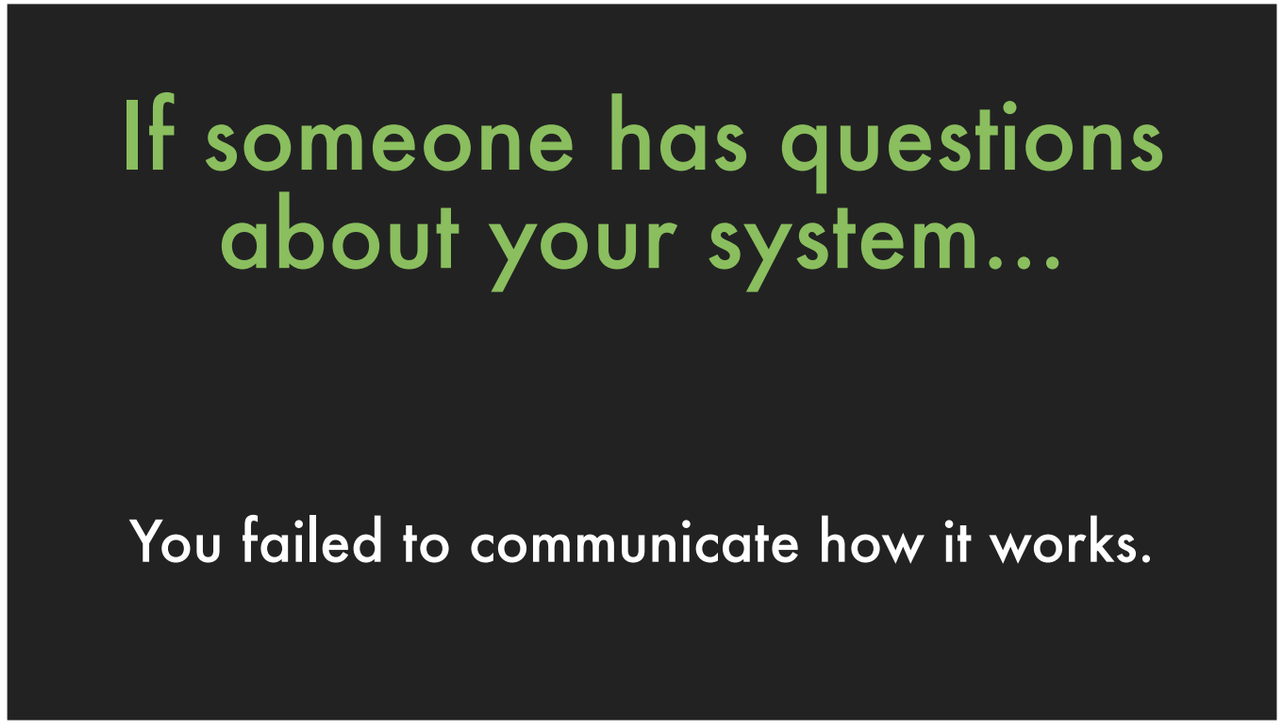
someone can't understand your system?

- when your system has failed?
- when someone has questions about your system?
- when someone can't understand your system?
- Do you get angry? Do you get defensive? Do you blame others?
- This is where ownership gets extreme, because the focus shifts from what we naturally point to, others, and onto ourselves.
-

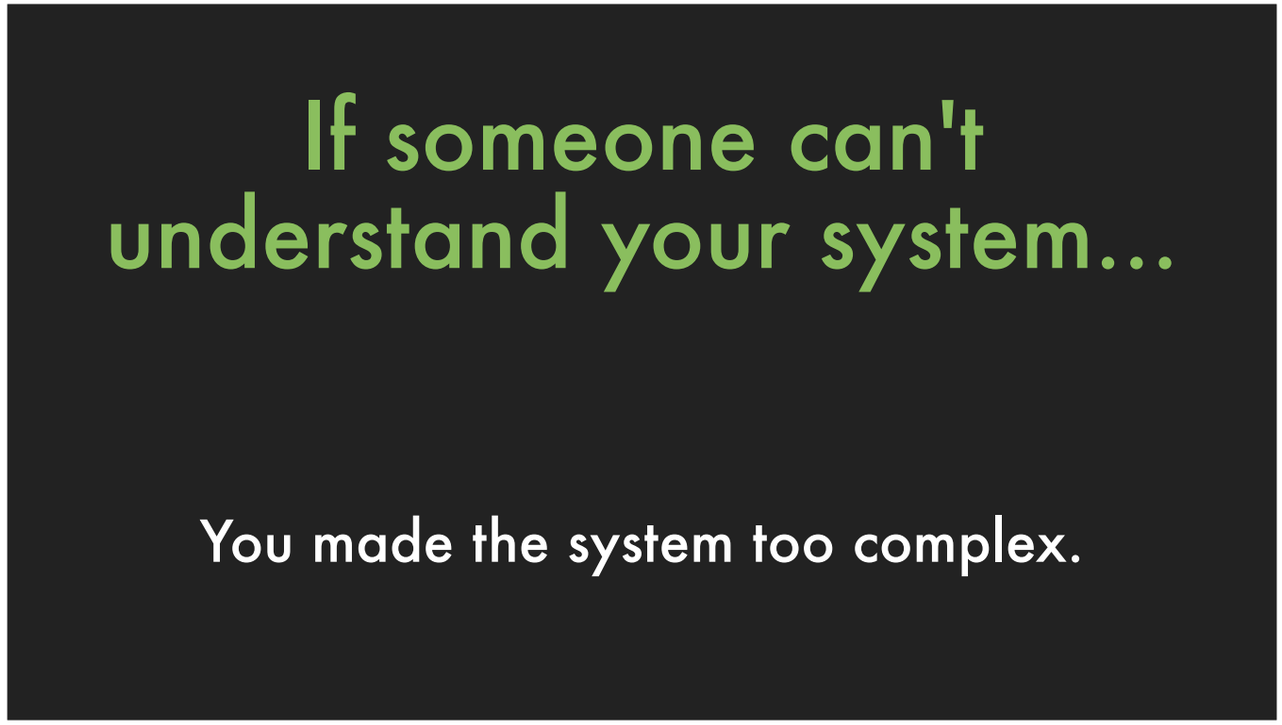



- So, If your system has failed
- ***
- you failed to protect the system.
- Say your script breaks because someone forgot a flag. Don’t say: “You should have known better”, ask yourself: “Can I add a protection in the code? Could I have improved the documentation to make this less likely?”
- This is hard. This is hard because it requires you to drop your ego. This is hard because it requires you to be willing to admit to imperfection.
- But wait.
- If you are feeling argumentative, about this or one of my later examples, if you feel like what I just said isn’t fair or is missing nuance, then please,


- relax your ego
- or maybe you need help working on some boundaries.
- Once you let go of your ego, that is your pride or self importance, you will produce better results and build better relationships with your colleagues.
- You will find it easier to listen and easier think objectively about your own technical system when you aren't worried about protecting yourself at the cost of others.
- But I do need to point out that some degree of self-protection is a natural response for anyone who has come from a past of being taken advantage of.
- Working with your manager or a trusted friend or even a professional therapist or psychologist can help you sort out what sort of reaction is going on here depending on how difficult you find it to respond with compassion.
- Anyway, let’s get back to those reactive questions:
-




- If someone has questions about your system,
- ***
- you failed to communicate how it works.
- Did you miss a chance to train them? Does something need to be renamed in the code? Did you miss a chance to document it? Maybe now is a really good opportunity to go and write that documentation.
-




- if someone can't understand your system,
- ***
- you made the system too complex.
- At the very least, don't insult the engineer by telling them they're to stupid to understand it. But equally important, and something I've been guilty of myself, is to not brag on the complexity of your creation.
- "Yeah, it's really quite special how I added those three levels of cache in there"
- No, take a moment and see it for what it is in their eyes: An overly complicated mess. Again you have to have some humility and be willing to ask “Did I really need to put those three levels of cache in there? Are they actually working like I thought they were going to?”
-

# Tempering "Extreme" with specific direction

- It is all well and good to ask ourselves these questions, but I want to make sure we don't make everything extreme, so lets regulate it and to make sure you know what should and shouldn't be extreme.
-

# Problems

~~You achieve having no problems.~~

You aim to avoid repeat problems.

- Problems.
- Once you've finally embraced a lifestyle of extreme ownership, all your problems are going to go away right?
- ***
- Of course not! So we won't be extreme in that regard, but instead we'll be extreme in our drive to avoid repeat problems.
- I mentioned that script breaking a moment ago. So if it fails once, you probably just chalk it up to a fluke right? But when it keeps on tripping alarms, then you notice and increase the priority of fixing it.
-

# Attitude

~~Know and teach on everything.~~

Learn from everything.

- But how are you going to feel about the intern pointing out that problem in your script?
- Extremely angry? Extremely annoyed?
- ***
- No, you must not be a know-it-all. You must have the humility to learn from everything and everyone.
-

# Boundaries

~~You have no boundaries.~~

You communicate your boundaries.

- But then the Mr-Sassy-Has-No-Patience-but-is-highly-respected-anyway on your team comes along and demands that you fix the bug right now.
- So do you work overtime to get it fixed?
- ***
- No! You don't tell them to shove off like maybe you really would like to, but you do tell them where your boundaries are.
- Maybe its "I agree that's a painful bug but I'm getting off soon and my manager has me committed to this project" or "I can get to that next sprint". You may even be able to add "If I rememeber, the script can be fixed with a single if statement, but I don't have the capacity to test it right now. I'd be happy to review your change if you could make it and do the testing".

# Example 1
# The System Crash

- Lets walk through a few more examples.

- Someone puts an incorrect combination of arguments into your tool and the system crashes.
- Do you say:
- "Well, if you would have just read the docs, this wouldn't have happened."
- or
- "Ah, we could add a linter rule that prevents you from hurting yourself here"

# Example 2
# The Meeting

- Later, You get invited to a meeting about database migrations.
- Do you say:
- "Why am I wasting my time with another meeting?"
- or
- "I looked over the agenda and it looks like you are inviting me to talk about migrations. Did you know that we have a document covering how to build and run those here?"

# Example 3
# The New Guy

- Later, The new guy asks how to store a file and if that's what S3 is.
- Do you say:
- "Yeah. Its named 'Simple Storage Service'. Go read up on it, it's not hard"
- or
- "Yeah, thats right. Let me set up some time to meet tomorrow so we can cover how it gets used here in the company. In the mean time, here's a doc that may be helpful to skim"

# Example 4
# The TPS Reports

- Finally, Your boss walks by and says "Hey I hate logging into that new metrics tool. Can you run the report I need again this week?"
- Maybe you could say "Sure, and hey, what's makes it difficult to log into? We bought the tool because you were excited about how effortless it made getting those reports."
- Then they say "Yeah, that was before I knew it was going to make me use a longer and more complicated password than I use for anything else. I can never remember what it is"
- And that's when I realized a C-Level leader in the company has been using the same one password for every account and that it consisted of a single word with a single digit.

- I taught them how to use a real password manager.

Extreme Ownership is to own everything related to the success of your mission.

For a small team, make sharing and simplifying an important part of your work.

Be humble.
Learn from mistakes.
Learn from others.
Communicate boundaries.

---

# THANK YOU

Danny Kirchmeier
Infrastructure Security Engineer @ Outschool
danny@outschool.com danny@kirchmeier.us