Can Infrastructure as Code apply to Bare Metal?

LISA 2021

Rob Hirschfeld (@zehicle)
CEO & Co-Founder, RackN

YES! Let's IaC some Bare Metal!

Lessons from 3+ years of IaC w/ Digital Rebar
LISA 2021

Rob Hirschfeld (@zehicle)
CEO & Co-Founder, RackN

Sequel to my 2020 SRECon PXE talk

Deep Diving into

- Netboot
- Image Deploy
- PXE / iPXE
- Cloud Init
- Secure Boot
- BMC Boot

What is Infrastructure as Code (IaC?)

Minimally, **Automation** in **Source Code**

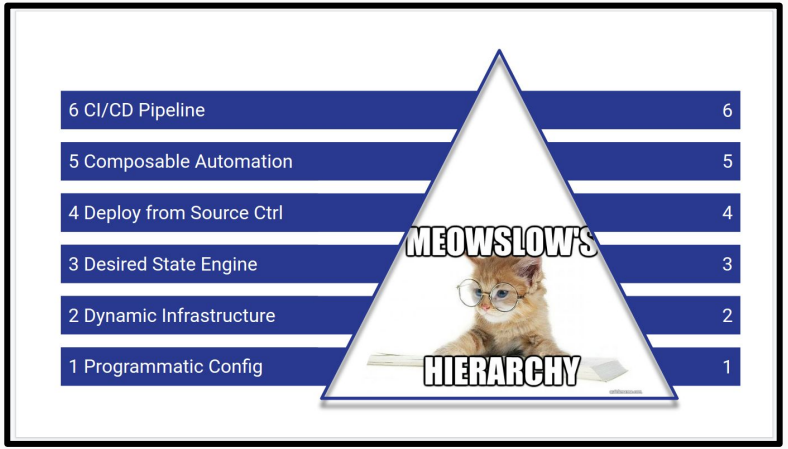
A Degree of **Immutability**

Aware of **Current State** (Reconcilier)

Seeking a **Desired State** (GitOps)

Collaboration and Reuse

Ideally, use of an Infrastructure **Pipeline**



From my “Aspiring to IaC” Talk

So what's new about IaC?

Why not call this DevOps?
Or SRE?
Or Run Book Automation?

It's not a job or product category, it's an automation approach.

IMHO, it's about making Operations more Developer like.

Image: Cameron Rainey from Pexels

Selling IaC to your boss?

Yes, it's cool tech and it's saves time/money/effort, but…

Selling IaC to business people is about **RISK REDUCTION**

- COMPLIANCE RISK IaC is more Transparent and Auditable
- DELIVERY RISK IaC is more Repeatable
- PEOPLE RISK IaC is more Collaborative

You'll need money, time and political cover to implement IaC.

Why is Bare Metal so Hard?

We can use these lessons from bare metal and cloud!

Even if cloud has better APIs, the challenges really are the same.

And the cloud process strategies also improve bare metal.

Have to deal with what you find

APIs can’t recable the box

Necessary Complexity

Abstractions are not always useful

Multiple APIs

Juggle DHCP, TFTP, HTTP, DNS, SSH...

Provisioning *and* Configuration

Work is outside *and* inside the server

Automation in Source Code

Image: Swapnil Deshpandey from Pexels

Your Ops tools should expect SCM

Git everything you use to build!

- Bash scripts, Ansible Playbooks, Terraform Plans
- Define and extract out the inputs!
- Use Git Checksums, Tags and Versions

And for Bare Metal?

- Kick Start, Preseed, Weasel

Immutability

Build with Less Touch, More Love

Make it fast and easy to recreate instances

- Use image based deployment
- Some post-config is OK, but keep it minimal
- Replace, don’t patch

And for Bare Metal?

- When Resetting/Repaving systems include FIRMWARE update!

Current State (Reconcilier)

Do Not Trust any Single Source of Truth!

Tracking State is central to IaC

- All parts of your pipeline should accept EXTERNAL updates
- Expect state to drift outside of controls
- Automation should STOP if state does not match

And for Bare Metal?

- Assume everything changed: check, verify, act, confirm

Desired State (bleh, GitOps)

Image: Ficky on Pexels

More Declarative! (And Less Imperative) ***link***

Set the end goal then let the automation do the work

- End goal description should be human understandable
- Beware of broken tool handoffs
- Beware of single source of truth mythology

And for Bare Metal?

- Be prepared for multiple reboots and many many different APIs

Infrastructure Pipeline

Image: Shukhrat Umarov on Pexels

No operation happens in isolation

Automation steps should connect together

- Require silos to have APIs that enable chaining
- Beware of systems that rely on their own state
- Bias towards event-based & observable systems

And for Bare Metal?

- Think cloud! Use Immutability & Repaving aggressively

Collaboration and Reuse

Build your automation so others can reuse it!

Collaboration is the soul of IaC

- Did you write some documentation? No?! Go back 2 squares.
- Decompose automation into reusable blocks
- Declare your variables! Seriously, NO AD HOC INPUT/OUTPUT.

And for Bare Metal?

- If you are coding here, you are not adding any value. Stop.

Let’s make this work for Bare Metal!

Topics:

1. Physical Control Challenges
2. Physical Reuse Challenges
3. Making Operations Immutable
4. Building Infrastructure Pipelines

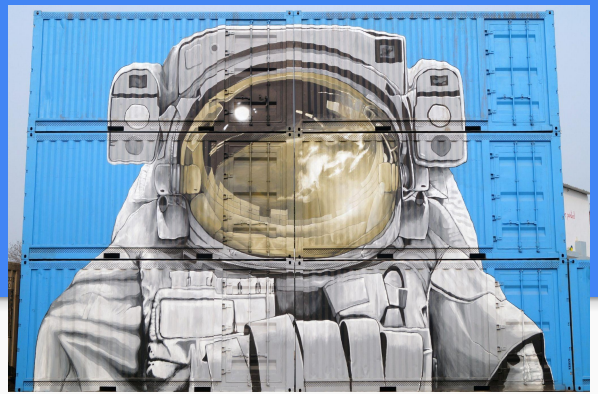
And yet, none of these points are really bare metal specific!

They apply for any heterogeneous system, so ALL systems.

Image: Myles

Physical Control Challenges

Image: Pixabay via Pexels



Cloud APIs are BEAUTIFUL, but...

“Just give me a server” cannot be that simple in bare metal

- Limits of cloud and container APIs for infrastructure
- Balance of abstractions in APIs vs need to control
  - Java Hibernate vs Rails ActiveRecord
- Dealing with limitations of existing infrastructure
  - Don’t get frustrated when it happens
  - Take time to solve

Physical Reuse Challenges

Image: Leonid Danilov from Pexels

- Why is firmware so hard?
  - Really! I'm looking at you hardware OEMs
  - But Redfish? Sorry, still hard.
- Dealing with heterogeneity in systems
  - Even if you are single vendor, there's variation!
  - It's normal - build it as an expected thing
  - Do not assume everything is the same
- Finding and sharing reusable components
  - Get out of the business of firmware if you can!
  - Reuse, Reuse, Reuse - you'll have incent this behavior

Making Operations Immutable

- Immutable image deployments
  - They are FASTER
  - They are more resilient
  - They are more secure
  - This is low hanging fruit!
- Think in RESETs, not provisions
  - More cloud-like behavior
  - Discourages single run automation

Image: Shukhrat Umarov on Pexels

Building Infrastructure Pipelines

Pipelines ARE THE KEY to IaC!

- Provisioning AND Configuration
  - Needs to work in tandem
  - Should be a standard process
  - Do not assume they are layered - they are really MIXED
- Coordinating operations across systems
  - Every system needs to participate in the process
  - CI/CD Development Mantra translates to Infrastructure
  - Pipelines change State (no single source of truth!)

Next level challenges

Image: Colourblind Bob

Scaling Up IaC

Better to handle variation
Deal with change over time

Distributed Infrastructure

IaC = easier to replicate sites!
Deal with change over time

Automation Portability

Portability is key for reusability
This is how DEVELOPERS work!

Demo!

What does an Infrastructure Pipeline look like action?

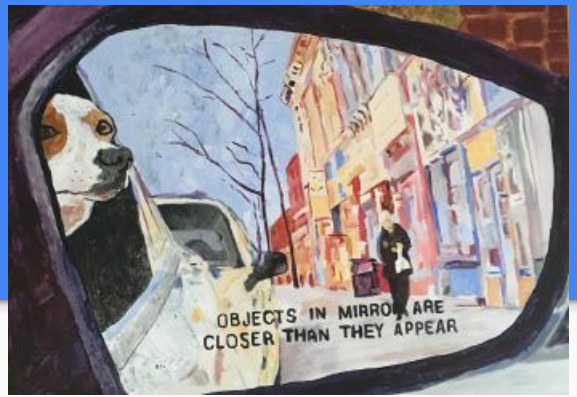
We've been collaborating with our enterprise customers to build a "Universal Workflow" in Digital Rebar.

Out of the box, it includes *all* the process steps required to fully deploy any platform in our library.

Operators select a target profile and workflow chooses the right stages to deliver that target.

To review

Image: Jimmy Tidwell via Pinterest



What can you do to advance IaC?

1. Adding Infrastructure to CI/CD
2. Figure out what you safely put in source control
3. Improving control *but without* customization
4. Rethinking how you store system state (no islands)
5. And legacy is not bad! Get over that.

Thank you!

We’re sponsors, please come say “hey” to the team that the booth!

Rob Hirschfeld (@zehicle)
CEO & Co-Founder, RackN

Image: Myles via Flickr