Generative DevOps?

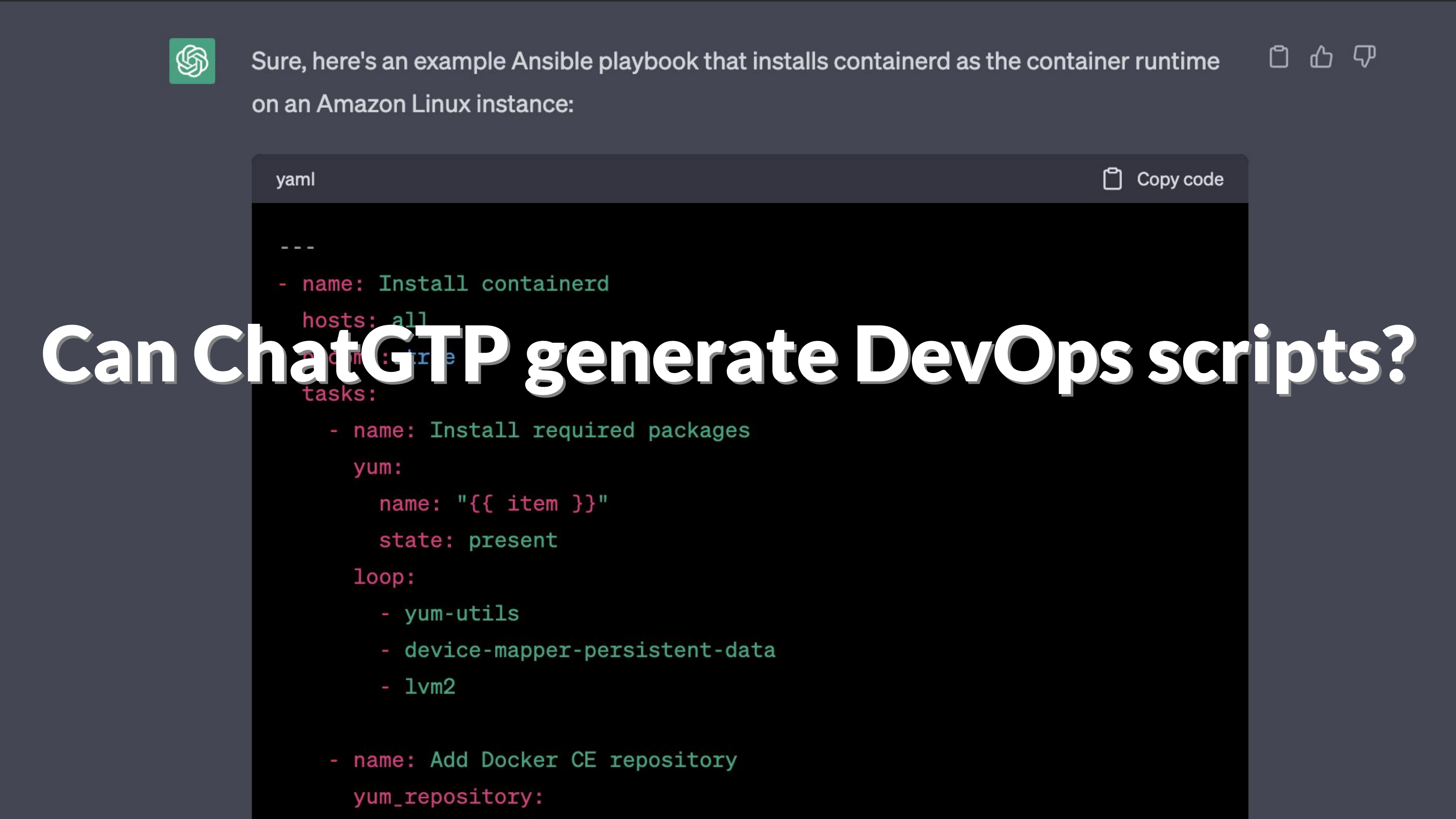
Better DevOps with Generative AI

Practical insights on harnessing LLMs
to serve as expert advisors
instead of script generators

Rob Hirschfeld, CEO of RackN

LinkedIn @RobHirschfeld
Others @Zehicle

Can ChatGTP generate DevOps scripts?



Sure, here's an example Ansible playbook that installs containerd as the container runtime on an Amazon Linux instance:

```yaml
---
- name: Install containerd
  hosts: all
  become: true
  tasks:
    - name: Install required packages
      yum:
        name: "{{ item }}"
        state: present
      loop:
        - yum-utils
        - device-mapper-persistent-data
        - lvm2

    - name: Add Docker CE repository
      yum_repository:
```

“ChatGPT, make me a cloud!”

```
Clouds are formed when warm air rises
and expands …
```

“I meant a computer cloud”

```
Step 1. Sign Up for AWS …
```

“use terraform”

```
Create a Terraform Configuration File
…
[Minimal main.tf created]
```

Yikes! Is this `main.tf` OK to use?

```
provider "aws" {
    region = "us-west-1" # You can choose your preferred region
}
resource "aws_instance" "example" {
    ami           = "ami-0c55b159cbfafe1f0" # This is an Amazon Linux 2 AMI, might need to be updated
    instance_type = "t2.micro"
    tags = {
        Name = "terraform-example"
    }
}
```

Yikes! Is this `main.tf` OK to use?

```
provider "aws" {
    region = "us-west-1" # You can choose your preferred region
}
resource "aws_instance" "example" {
    ami           = "ami-0c55b159cbfafe1f0" # This is an Amazon Linux 2 AMI, might need to be updated
    instance_type = "t2.micro"
    tags = {
        Name = "terraform-example"
    }
}
```

Assumed AWS

Popular Region

Good naming?

Credentials?!

Is this the right AMI?

Why this size?

Potential name issue

What about networking, SSH, storage, etc

LLM offers appearance of expertise, but without knowing...

Your Policies

- Using AWS?
- Which Region?
- Which O/S?
- Naming Convention?
- Required Setup?

LLM offers appearance of expertise, but without knowing...

Your Policies

- Using AWS?
- Which Region?
- Which O/S?
- Naming Convention?
- Required Setup?

Syntax Rules

- AMI alignment?
- Allowed config?
- Will it compile?
- Does it work?
- Well designed?
- Does it use the right version?

LLM offers appearance of expertise, but without knowing...

Your Policies

- Using AWS?
- Which Region?
- Which O/S?
- Naming Convention?
- Required Setup?

Syntax Rules

- AMI alignment?
- Allowed config?
- Will it compile?
- Does it work?
- Well designed?
- Does it use the right version?

Ops Context

- Is this right for your infra?
- Is this secure?
- Who will maintain it?
- Who will pay for the instance?

RackN

Photo by Harrison Haines:

"We Already Have Too Much Automation"

VP of Infrastructure, Top US Bank

If so, LLMs will just add to the existing a dumpster fire!

How can we fix automation?

RackN

Who is Rob Hirschfeld?
Who is RackN?

RackN

We are Making Better Operators

Infrastructure as Code Automation Software
DevOps Scaffolding (aka Digital Rebar)

We empower operators

What’s holding operators back?

SURPLUS OF

- Equipment types
- Equipment vendors
- Documentation
- Tooling silos
- Architecture

Photo by Riccardo Pitzalis

What's holding operators back?

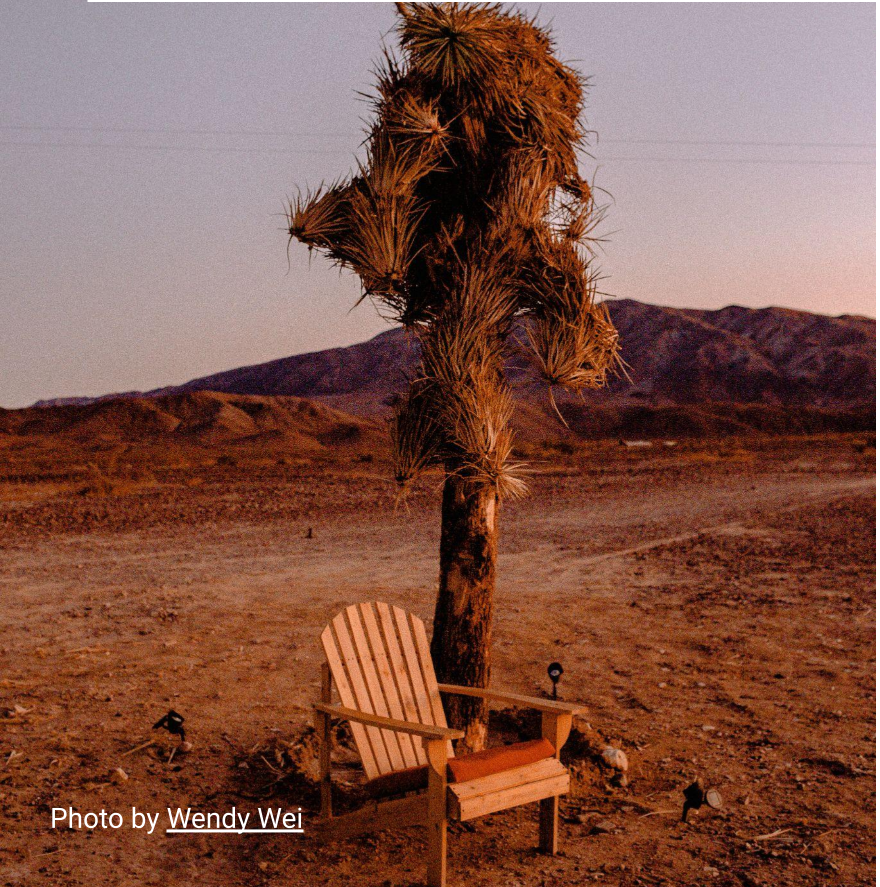


Photo by Wendy Wei

LACK OF

- ⦿ Reusability
- ⦿ Validation & Linting
- ⦿ Backwards compatibility
- ⦿ Resilience
- ⦿ Shared state

RackN

How does LLM help operators?

Expected :

Write scripts

Refactor scripts

How does LLM help operators?

Expected :

Write scripts
Refactor scripts

Surprised !

Quality of output
Translate format
Write Summaries

How does LLM help operators?

Expected :

Write scripts
Refactor scripts

Surprised !

Quality of output
Translate format
Write Summaries

Delighted $

Explaining scripts
Finding alternatives
Planning operations

How does LLM help operators?

Expected :

Write scripts
Refactor scripts

Surprised !

Quality of output
Translate format
Write Summaries

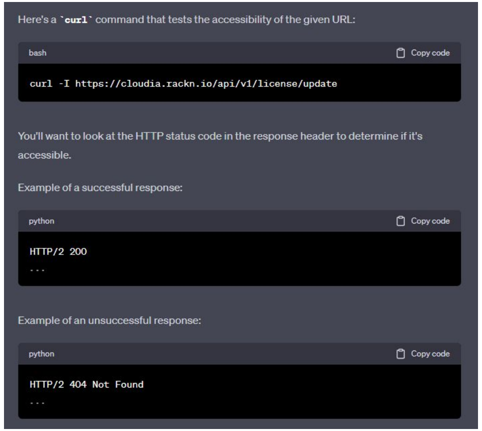
Delighted $

Explaining scripts
Finding alternatives
Planning operations

If you already know DevOps, it enhances your productivity

Fast code snippets

Test URL?



Here's a `curl` command that tests the accessibility of the given URL:

```bash
curl -I https://cloudia.rackn.io/api/v1/license/update
```

You'll want to look at the HTTP status code in the response header to determine if it's accessible.

Example of a successful response:

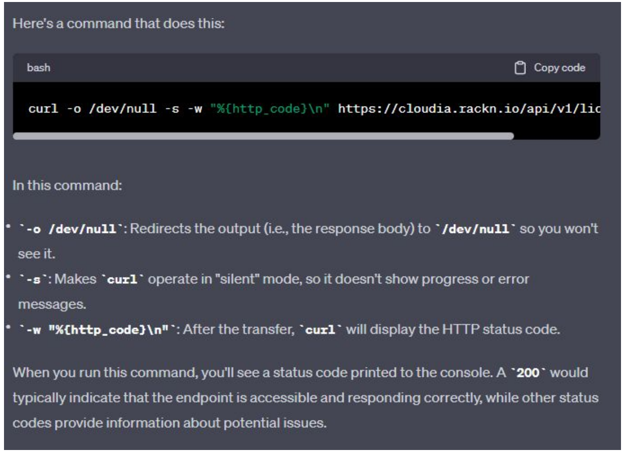
```python
HTTP/2 200
...
```

Example of an unsuccessful response:

```python
HTTP/2 404 Not Found
...
```

Tested...
did not work!?!

Second Try



Here's a command that does this:

```bash
curl -o /dev/null -s -w "%{http_code}\n" https://cloudia.rackn.io/api/v1/lic
```

In this command:

- `-o /dev/null`: Redirects the output (i.e., the response body) to `/dev/null` so you won't see it.
- `-s`: Makes `curl` operate in "silent" mode, so it doesn't show progress or error messages.
- `-w "%{http_code}\n"`: After the transfer, `curl` will display the HTTP status code.

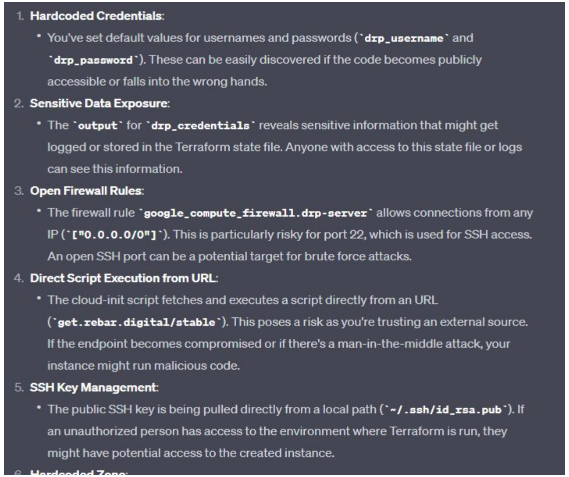
When you run this command, you'll see a status code printed to the console. A `200` would typically indicate that the endpoint is accessible and responding correctly, while other status codes provide information about potential issues.

Use for RegEx, Bash & Powershell!

RackN

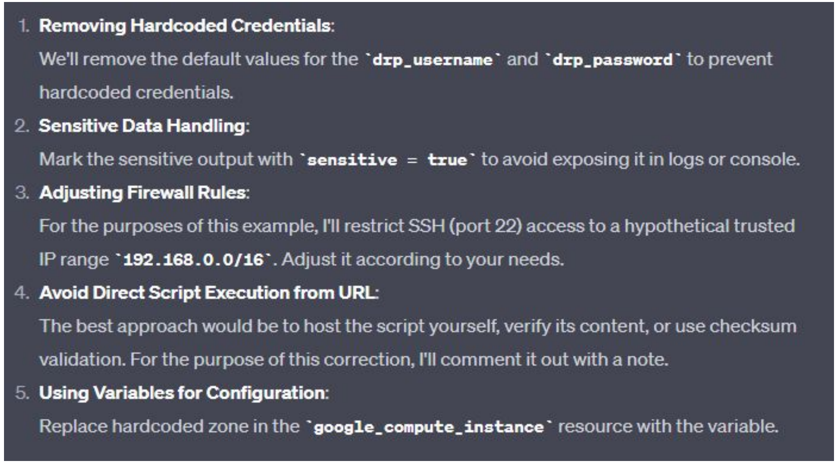
Code review and recommendations

Look at my TF



1. **Hardcoded Credentials**:
   - You've set default values for usernames and passwords (`` `drp_username` `` and `` `drp_password` ``). These can be easily discovered if the code becomes publicly accessible or falls into the wrong hands.
2. **Sensitive Data Exposure**:
   - The `` `output` `` for `` `drp_credentials` `` reveals sensitive information that might get logged or stored in the Terraform state file. Anyone with access to this state file or logs can see this information.
3. **Open Firewall Rules**:
   - The firewall rule `` `google_compute_firewall.drp-server` `` allows connections from any IP (`` `["0.0.0.0/0"]` ``). This is particularly risky for port 22, which is used for SSH access. An open SSH port can be a potential target for brute force attacks.
4. **Direct Script Execution from URL**:
   - The cloud-init script fetches and executes a script directly from an URL (`` `get.rebar.digital/stable` ``). This poses a risk as you're trusting an external source. If the endpoint becomes compromised or if there's a man-in-the-middle attack, your instance might run malicious code.
5. **SSH Key Management**:
   - The public SSH key is being pulled directly from a local path (`` `~/.ssh/id_rsa.pub` ``). If an unauthorized person has access to the environment where Terraform is run, they might have potential access to the created instance.
6. **Hardcoded Zone**:

Recommend Improvements

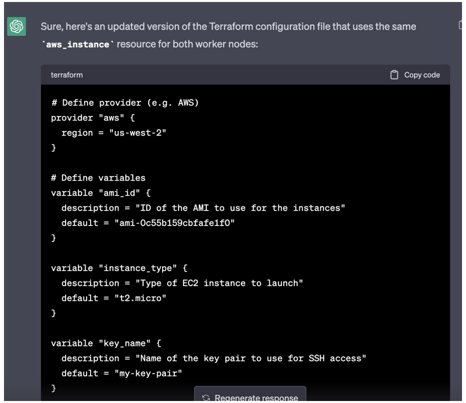


1. **Removing Hardcoded Credentials**:
   We'll remove the default values for the `` `drp_username` `` and `` `drp_password` `` to prevent hardcoded credentials.
2. **Sensitive Data Handling**:
   Mark the sensitive output with `` `sensitive = true` `` to avoid exposing it in logs or console.
3. **Adjusting Firewall Rules**:
   For the purposes of this example, I'll restrict SSH (port 22) access to a hypothetical trusted IP range `` `192.168.0.0/16` ``. Adjust it according to your needs.
4. **Avoid Direct Script Execution from URL**:
   The best approach would be to host the script yourself, verify its content, or use checksum validation. For the purpose of this correction, I'll comment it out with a note.
5. **Using Variables for Configuration**:
   Replace hardcoded zone in the `` `google_compute_instance` `` resource with the variable.

Do NOT accept anything on faith!

Refactoring is powerful

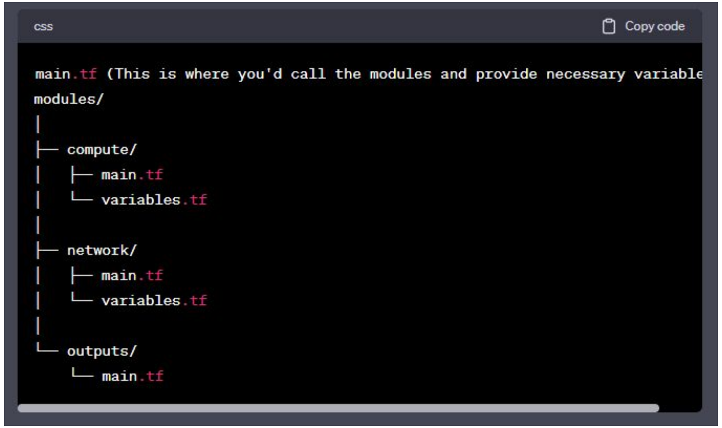
Terraform X



Sure, here's an updated version of the Terraform configuration file that uses the same `aws_instance` resource for both worker nodes:

```
# Define provider (e.g. AWS)
provider "aws" {
  region = "us-west-2"
}

# Define variables
variable "ami_id" {
  description = "ID of the AMI to use for the instances"
  default = "ami-0c55b159cbfafe1f0"
}

variable "instance_type" {
  description = "Type of EC2 instance to launch"
  default = "t2.micro"
}

variable "key_name" {
  description = "Name of the key pair to use for SSH access"
  default = "my-key-pair"
}
```

Convert into Modules

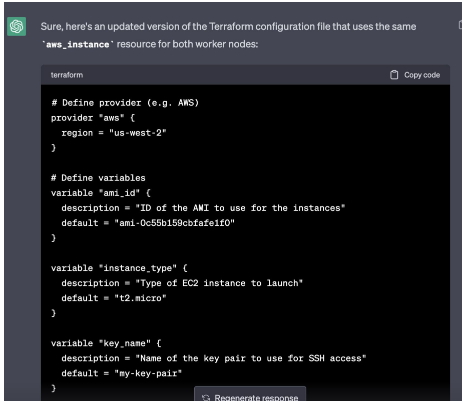


```
main.tf (This is where you'd call the modules and provide necessary variable
modules/
|
├── compute/
|   ├── main.tf
|   └── variables.tf
|
├── network/
|   ├── main.tf
|   └── variables.tf
|
└── outputs/
    └── main.tf
```

does NOT reuse existing modules!

Replatforming is game changing

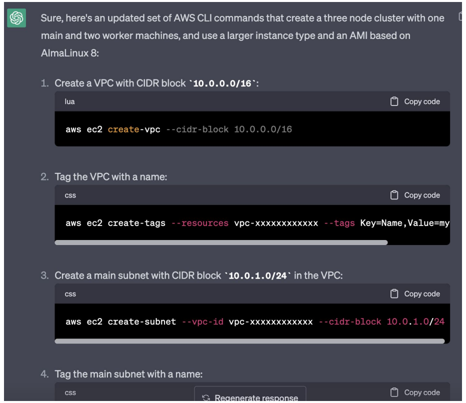
Terraform X



Sure, here's an updated version of the Terraform configuration file that uses the same `aws_instance` resource for both worker nodes:

```terraform
# Define provider (e.g. AWS)
provider "aws" {
  region = "us-west-2"
}

# Define variables
variable "ami_id" {
  description = "ID of the AMI to use for the instances"
  default = "ami-0c55b159cbfafe1f0"
}

variable "instance_type" {
  description = "Type of EC2 instance to launch"
  default = "t2.micro"
}

variable "key_name" {
  description = "Name of the key pair to use for SSH access"
  default = "my-key-pair"
}
```

into CLI & Bash



Sure, here's an updated set of AWS CLI commands that create a three node cluster with one main and two worker machines, and use a larger instance type and an AMI based on AlmaLinux 8:

1. Create a VPC with CIDR block `10.0.0.0/16`:

```lua
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

2. Tag the VPC with a name:

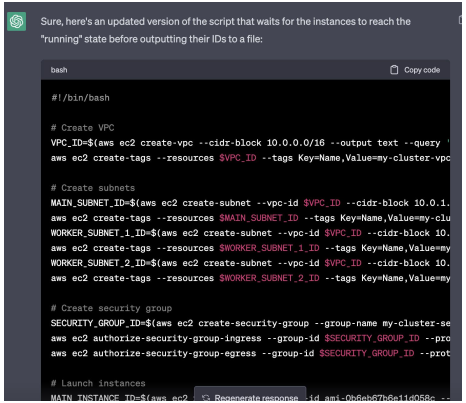
```css
aws ec2 create-tags --resources vpc-xxxxxxxxxxxx --tags Key=Name,Value=my
```

3. Create a main subnet with CIDR block `10.0.1.0/24` in the VPC:

```css
aws ec2 create-subnet --vpc-id vpc-xxxxxxxxxxxx --cidr-block 10.0.1.0/24
```

4. Tag the main subnet with a name:

then Templatize



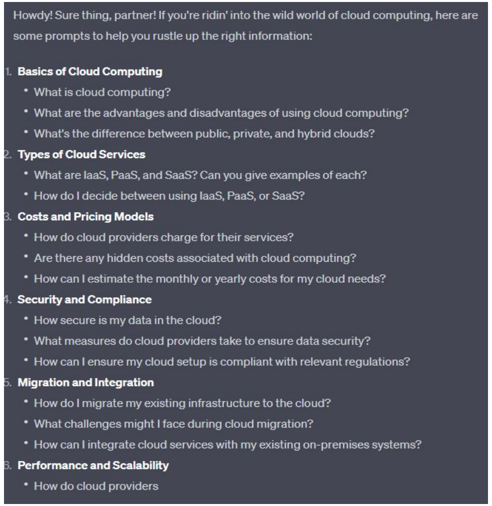
Sure, here's an updated version of the script that waits for the instances to reach the "running" state before outputting their IDs to a file:

```bash
#!/bin/bash

# Create VPC
VPC_ID=$(aws ec2 create-vpc --cidr-block 10.0.0.0/16 --output text --query '
aws ec2 create-tags --resources $VPC_ID --tags Key=Name,Value=my-cluster-vpc

# Create subnets
MAIN_SUBNET_ID=$(aws ec2 create-subnet --vpc-id $VPC_ID --cidr-block 10.0.1.
aws ec2 create-tags --resources $MAIN_SUBNET_ID --tags Key=Name,Value=my-clu
WORKER_SUBNET_1_ID=$(aws ec2 create-subnet --vpc-id $VPC_ID --cidr-block 10.
aws ec2 create-tags --resources $WORKER_SUBNET_1_ID --tags Key=Name,Value=my
WORKER_SUBNET_2_ID=$(aws ec2 create-subnet --vpc-id $VPC_ID --cidr-block 10.
aws ec2 create-tags --resources $WORKER_SUBNET_2_ID --tags Key=Name,Value=my

# Create security group
SECURITY_GROUP_ID=$(aws ec2 create-security-group --group-name my-cluster-se
aws ec2 authorize-security-group-ingress --group-id $SECURITY_GROUP_ID --pro
aws ec2 authorize-security-group-egress --group-id $SECURITY_GROUP_ID --prot

# Launch instances
MAIN_INSTANCE_ID=$(aws ec2                            -id ami-0b6eb67b6e11d058c --
```

Tool lock-in disappears

Prompting and Tone matter!

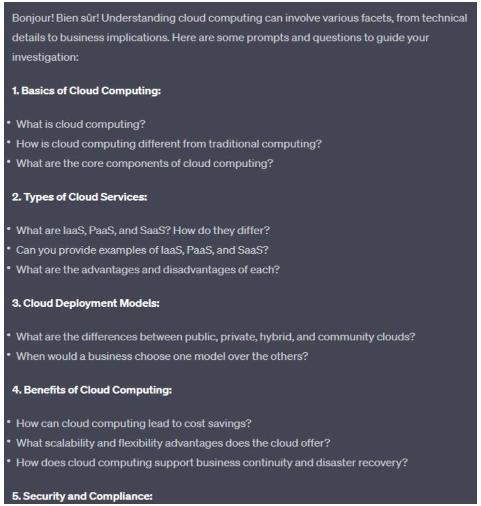
Cowboy



Howdy! Sure thing, partner! If you're ridin' into the wild world of cloud computing, here are some prompts to help you rustle up the right information:

1. **Basics of Cloud Computing**
   - What is cloud computing?
   - What are the advantages and disadvantages of using cloud computing?
   - What's the difference between public, private, and hybrid clouds?
2. **Types of Cloud Services**
   - What are IaaS, PaaS, and SaaS? Can you give examples of each?
   - How do I decide between using IaaS, PaaS, or SaaS?
3. **Costs and Pricing Models**
   - How do cloud providers charge for their services?
   - Are there any hidden costs associated with cloud computing?
   - How can I estimate the monthly or yearly costs for my cloud needs?
4. **Security and Compliance**
   - How secure is my data in the cloud?
   - What measures do cloud providers take to ensure data security?
   - How can I ensure my cloud setup is compliant with relevant regulations?
5. **Migration and Integration**
   - How do I migrate my existing infrastructure to the cloud?
   - What challenges might I face during cloud migration?
   - How can I integrate cloud services with my existing on-premises systems?
6. **Performance and Scalability**
   - How do cloud providers

French



Bonjour! Bien sûr! Understanding cloud computing can involve various facets, from technical details to business implications. Here are some prompts and questions to guide your investigation:

1. Basics of Cloud Computing:

- What is cloud computing?
- How is cloud computing different from traditional computing?
- What are the core components of cloud computing?

2. Types of Cloud Services:

- What are IaaS, PaaS, and SaaS? How do they differ?
- Can you provide examples of IaaS, PaaS, and SaaS?
- What are the advantages and disadvantages of each?

3. Cloud Deployment Models:

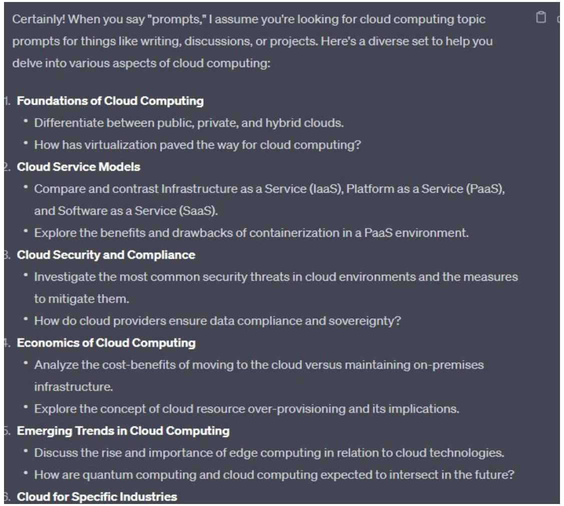
- What are the differences between public, private, hybrid, and community clouds?
- When would a business choose one model over the others?

4. Benefits of Cloud Computing:

- How can cloud computing lead to cost savings?
- What scalability and flexibility advantages does the cloud offer?
- How does cloud computing support business continuity and disaster recovery?

5. Security and Compliance:

Dude



Certainly! When you say "prompts," I assume you're looking for cloud computing topic prompts for things like writing, discussions, or projects. Here's a diverse set to help you delve into various aspects of cloud computing:

1. **Foundations of Cloud Computing**
   - Differentiate between public, private, and hybrid clouds.
   - How has virtualization paved the way for cloud computing?
2. **Cloud Service Models**
   - Compare and contrast Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).
   - Explore the benefits and drawbacks of containerization in a PaaS environment.
3. **Cloud Security and Compliance**
   - Investigate the most common security threats in cloud environments and the measures to mitigate them.
   - How do cloud providers ensure data compliance and sovereignty?
4. **Economics of Cloud Computing**
   - Analyze the cost-benefits of moving to the cloud versus maintaining on-premises infrastructure.
   - Explore the concept of cloud resource over-provisioning and its implications.
5. **Emerging Trends in Cloud Computing**
   - Discuss the rise and importance of edge computing in relation to cloud technologies.
   - How are quantum computing and cloud computing expected to intersect in the future?
6. **Cloud for Specific Industries**

It pays to be polite

Photo by Luiz Fernando

How is LLM doing this?

"Stochastic Parrots"

You need to

1. Prompting Templates
2. Vectorize your existing systems to baseline
3. Improve governance

RackN

Prompting & Prompt Templates

Setting the scene and tone will make a difference!

> ***Pro tip: "I need help from a DevOps professional who understands Linux"***

It remembers your conversation (good and bad), so you should build up the dialog and ask for changes and corrections.

It's smart of keep refining pre-wired framing information

Vectorize your existing systems to baseline

New tools allow you to “warm” your chat with existing information!

> ***Pro tip: Look for emerging tools that vectorize your data as input***

RackN is doing this with “RackNGPT” to help users learn Digital Rebar

But, watch out for sending sensitive information into the public domain!

Improve governance

You still need to OWN the results, so you need governance.

> ***Pro tip: Use LLMs as the first pass for review and check***

Don’t assume that generated code is safe, smart or maintainable!

Watch out for flooding your already overloaded human review systems with even more code. BUT, make sure you are getting reviews!

RackN

Zooming out...

Not Just Knowledge
This can be Expertise!

Photo by Brady Knoll

So What is Expertise?

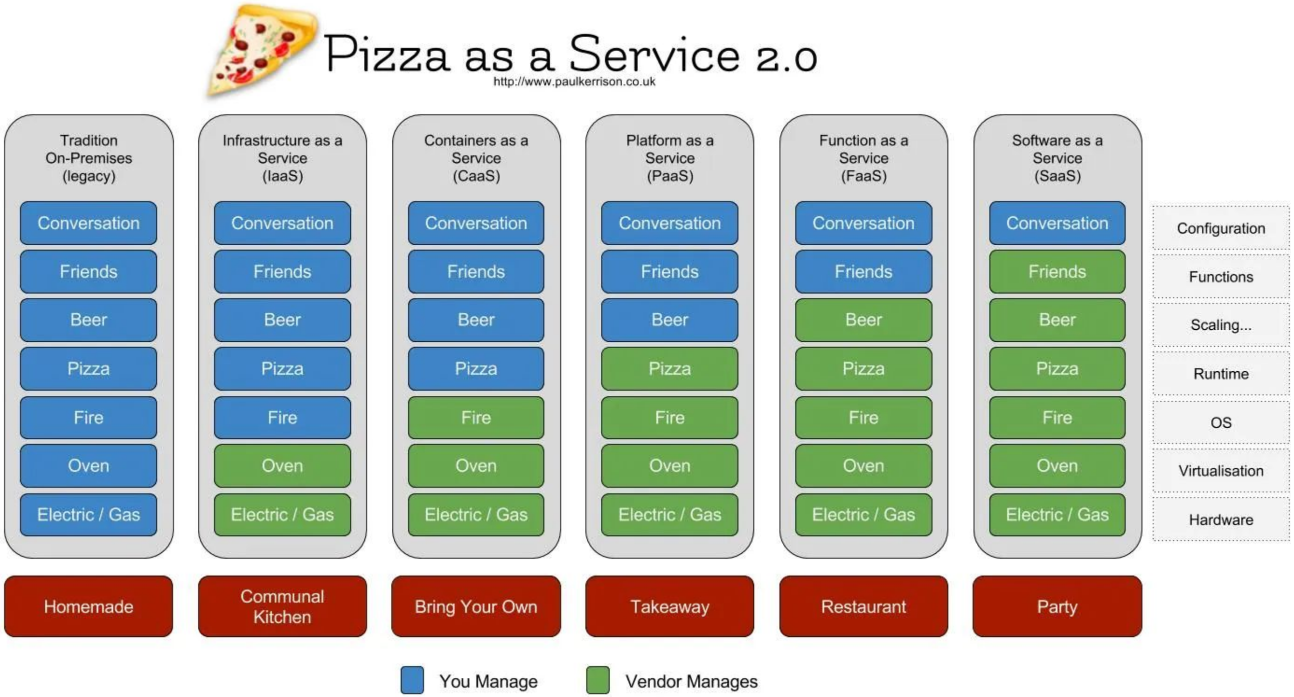
You vs. Cordon Bleu Chef

We have access to the ingredients and tools!

LLMs let you access deeper expertise
(if you use it correctly)

Photo by RDNE Stock project

Stacking “aaS” is WRONG
Ops Expertise > Infrastructure $



Pizza as a Service 2.0

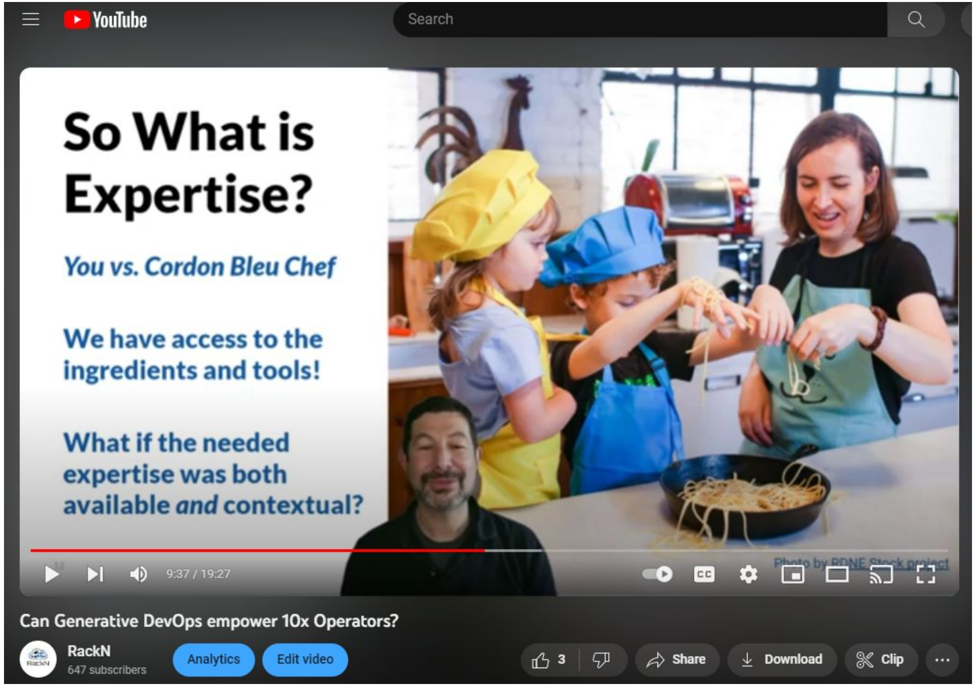
http://www.paulkerrison.co.uk

Tradition On-Premises (legacy)	Infrastructure as a Service (IaaS)	Containers as a Service (CaaS)	Platform as a Service (PaaS)	Function as a Service (FaaS)	Software as a Service (SaaS)	
🟦 Conversation	🟦 Conversation	🟦 Conversation	🟦 Conversation	🟦 Conversation	🟦 Conversation	Configuration
🟦 Friends	🟦 Friends	🟦 Friends	🟦 Friends	🟦 Friends	🟩 Friends	Functions
🟦 Beer	🟦 Beer	🟦 Beer	🟦 Beer	🟩 Beer	🟩 Beer	Scaling...
🟦 Pizza	🟦 Pizza	🟦 Pizza	🟩 Pizza	🟩 Pizza	🟩 Pizza	Runtime
🟦 Fire	🟦 Fire	🟩 Fire	🟩 Fire	🟩 Fire	🟩 Fire	OS
🟦 Oven	🟩 Oven	🟩 Oven	🟩 Oven	🟩 Oven	🟩 Oven	Virtualisation
🟦 Electric / Gas	🟩 Electric / Gas	🟩 Electric / Gas	🟩 Electric / Gas	🟩 Electric / Gas	🟩 Electric / Gas	Hardware
Homemade	Communal Kitchen	Bring Your Own	Takeaway	Restaurant	Party	

🟦 You Manage 🟩 Vendor Manages

Learn more about this…

[RackN “Can Generative DevOps empower 10x Operators?”]



So What is Expertise?

You vs. Cordon Bleu Chef

We have access to the ingredients and tools!

What if the needed expertise was both available *and* contextual?

Photo by RDNE Stock project

Can Generative DevOps empower 10x Operators?

RackN
647 subscribers

What can LLMs help fix?

SURPLUS OF

- 💗 Documentation
- 💗 Tooling silos
- 💪 Equipment types
- 💪 Equipment vendors
- 👀 Architecture

LACK OF

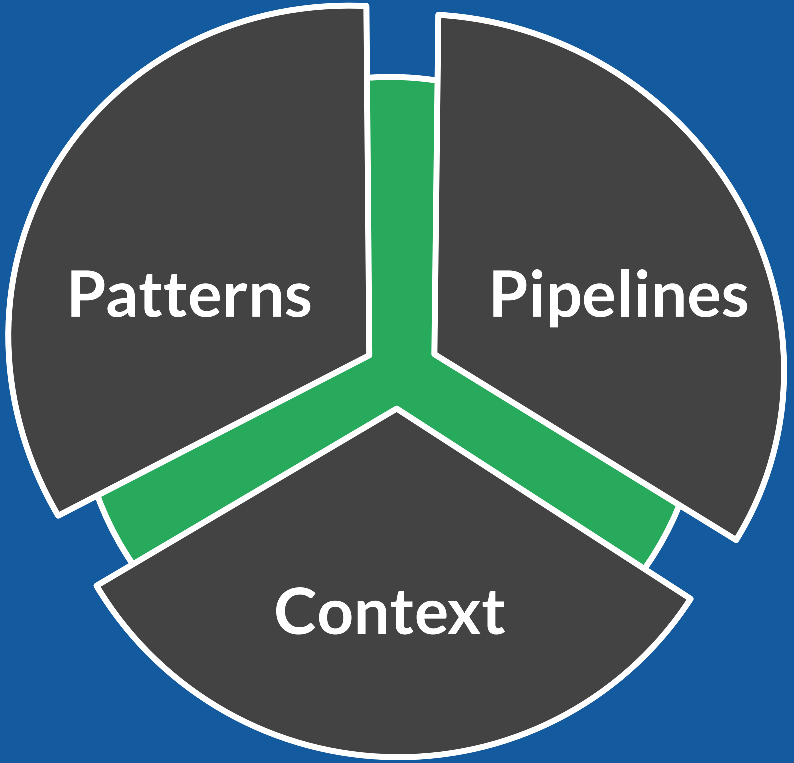
- 💗 Reusability
- 💗 Validation & Linting
- 💪 Backwards compatibility
- 💪 Resilience
- 👀 Shared state

RackN

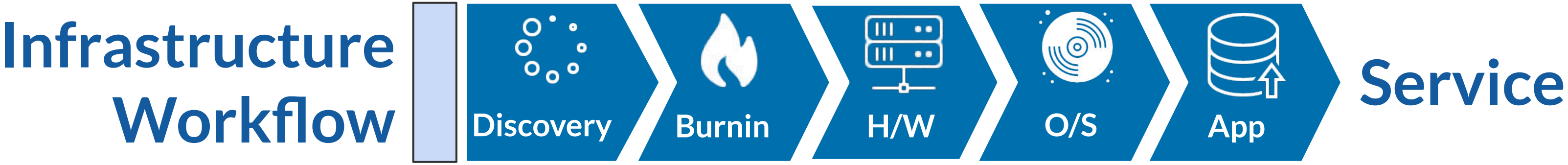
Operations is *not* just scripts

Systems Require:

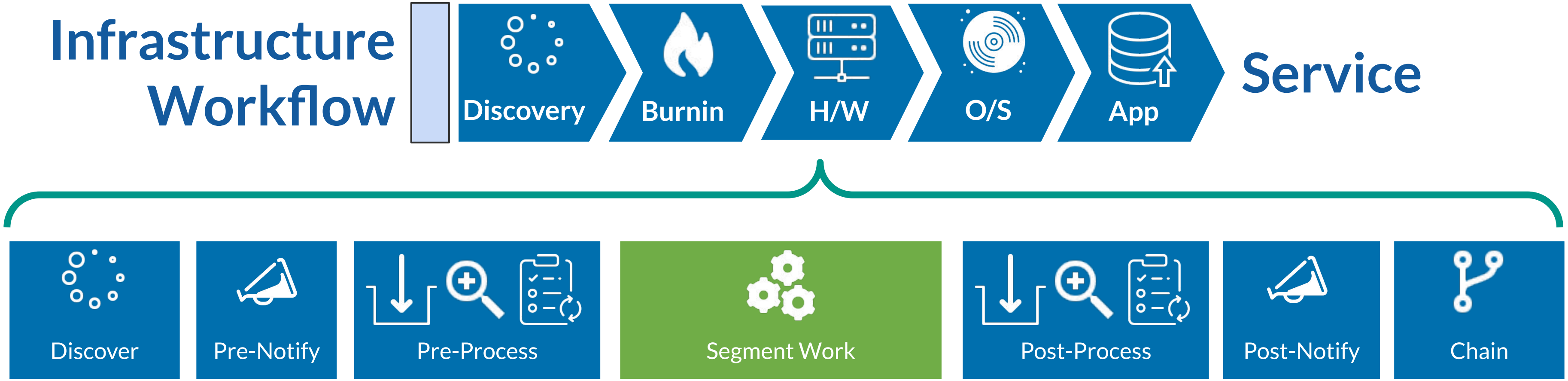
1. Architectural Patterns
2. Shared State Pipelines
3. Expertise in Context



RackN Composable Workflow



RackN Composable Workflow



Infrastructure Workflow

Discovery

Burnin

H/W

O/S

App

Service

Discover

Pre-Notify

Pre-Process

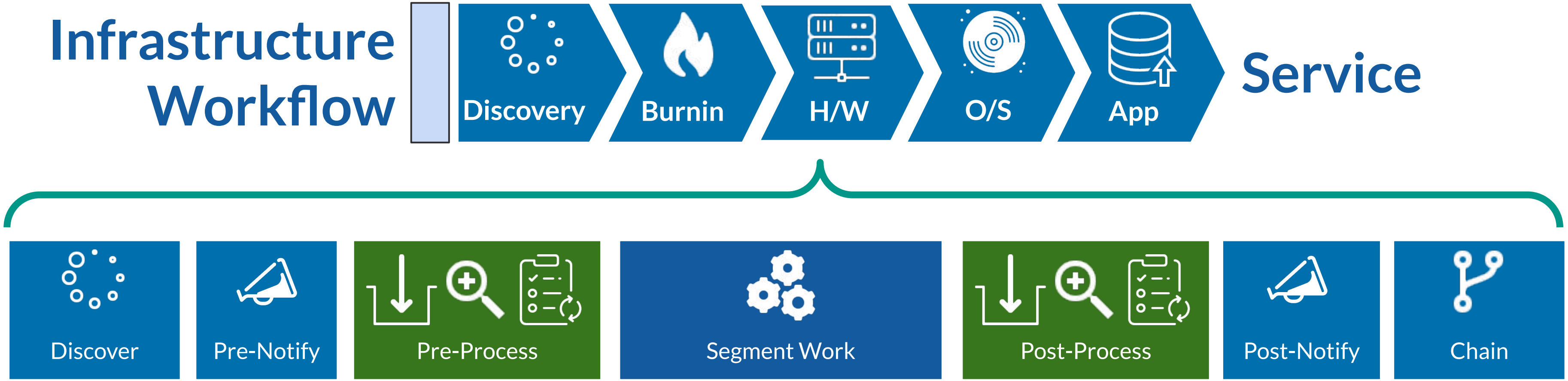
Segment Work

Post-Process

Post-Notify

Chain

RackN Composable Workflow



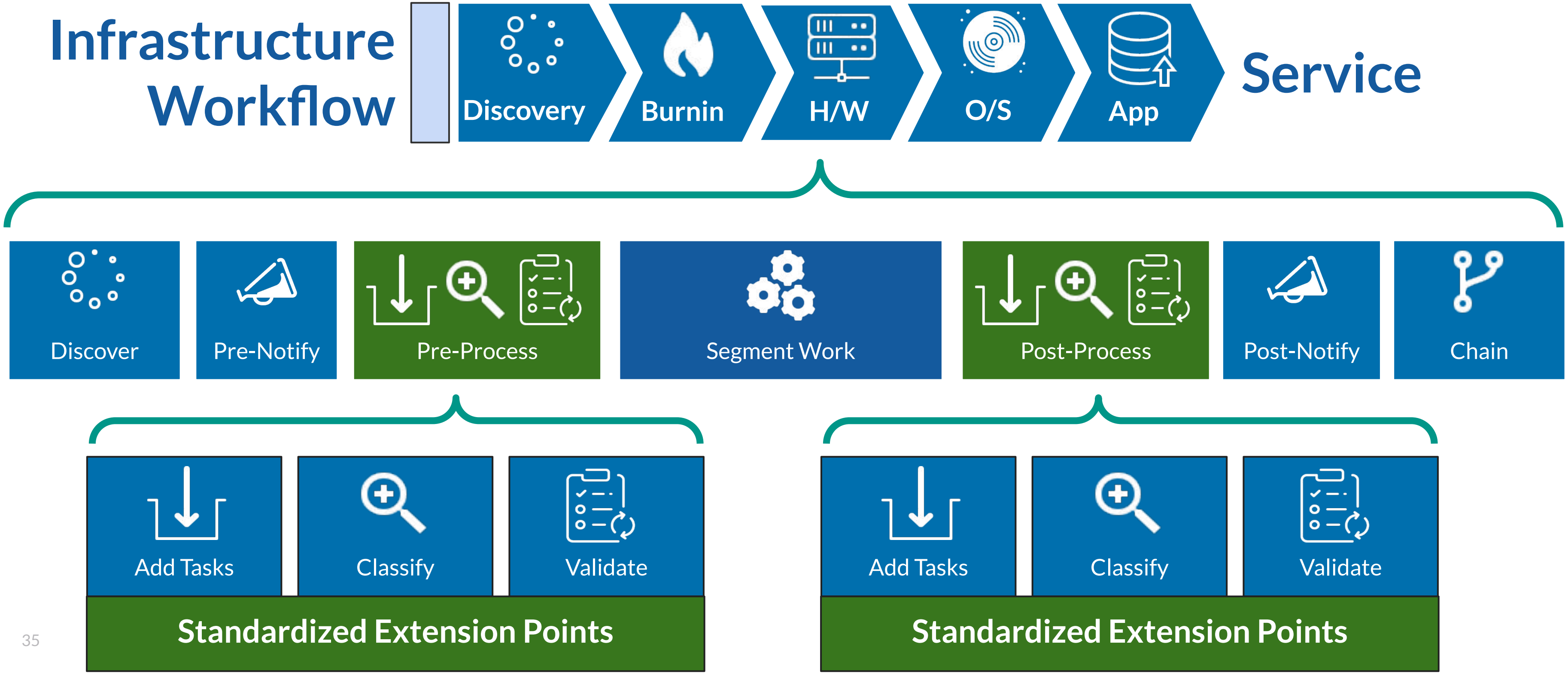
Infrastructure Workflow

Discovery → Burnin → H/W → O/S → App

Service

Discover | Pre-Notify | Pre-Process | Segment Work | Post-Process | Post-Notify | Chain

RackN Composable Workflow



Infrastructure Workflow

Discovery

Burnin

H/W

O/S

App

Service

Discover

Pre-Notify

Pre-Process

Segment Work

Post-Process

Post-Notify

Chain

Add Tasks

Classify

Validate

Standardized Extension Points

Add Tasks

Classify

Validate

Standardized Extension Points

Photo by Tom Fisk

Use LLMs to Reduce Sprawl

Learn to use it! And..

1. Build prompting templates
2. Vectorize your existing systems to baseline
3. Improve governance

Questions?

Want to become 10x Operator?
Visit us at RackN.com

Rob Hirschfeld, CEO of RackN

@RobHirschfeld
@Zehicle