Ryan Parman

Cloud-native engineering leader with a focus on reliability, scalability, and security for the modern web.

Currently: Enterprise Architect, Cloud Center of Excellence at McGraw Hill.

Summary

Ryan Parman is a cloud-native engineering leader with a focus on reliability, scalability, and security for the modern web. As an engineering problem-solver with over 25 years of experience across technical leadership, software development, site reliability engineering, and cybersecurity, he understands how to listen, learn, adapt, and improve.

He was a founding member of the AWS SDK team; patented multifactor-authentication-as-a-service at WePay; helped define the CI, CD, and SRE disciplines at McGraw Hill; came up with the idea of "serverless, event-driven, responsive functions in the cloud" while at Amazon Web Services in 2010 (AWS Lambda); and much, much more.

Technical Skills and Software

While my experience and personal technical interests are broad, the following list is focused more on my interest in DevTools, DevOps, and SRE roles. I would be happy to share additional experience for other areas upon request.

> 📓 **NOTE**
>
> Each entry here has a *current* proficiency level (scale: Low, Med, High, Expert), as well as a directional arrow. An up-arrow (↑) means I'm actively working with them and my proficiency is likely to go **up** over time. A down-arrow (↓) means it's been a while since I've worked with it, and my proficiency is likely to go **down** over time unless I get a good refresher course.

- **Operating Systems:** macOS (Expert: ↑), CentOS Linux (High: ↓), Amazon Linux 2 (High: ↓), Amazon Linux 2023 (High: ↑), Alpine Linux (High: ↑), Windows (Med), Ubuntu Linux (Med: ↑).
- **Standard Software Engineering Toolbox:** Dependency injection, performance profiling, character encodings, Git, Linux, Makefiles, and other fundamentals (High: ↑); memorized algorithms (Low); memorized Big-O notation (Low; I never learned it formally).
- **Programming Languages:** Golang (High: ↑), Python (High: ↑), Bash (High: ↑), *Modern* PHP (High: ↓) (not the bad old PHP that everyone hates), Browser JavaScript (High: ↓), Node.js JavaScript (Medium: ↓), Ruby (Low: ↓). Starting to learn Swift, but am just scratching the surface.

- **Cloud Computing:** AWS (Organizations, EC2, RDS, S3, CloudFront, SQS, SNS, IAM, STS, CloudWatch Monitoring, CloudWatch Logs + Insights, Lambda, ECS-on-EC2, ECR, API Gateway, Auto-scaling, CloudTrail, Elastic Transcoder, ElastiCache, Route 53, ELB/ALB, ACM, SSM, Parameter Store) (mostly High/Expert: ↑), AWS SDKs + CLI (High: ↑), AWS Well-Architected Framework (High: ↑); Google Cloud's core infrastructure services (Low: ↓), Microsoft Azure (None: ↑)

- **Provisioning:** Terraform/OpenTofu (Expert: ↑), Terragrunt (Med/High: ↑), Packer (High), Ansible (Med: ↓), Vagrant (Med: ↓), writing custom Modules (Expert: ↑), writing custom providers with the Plugin Framework (Med: ↑).

- **API and Scalable System Design:** Understanding and designing highly-scalable, distributed systems for running web applications and web services (High: ↑); JSON-over-HTTP web service API design (High); GraphQL with Relay implementations (Med/High: ↑); Understand the difference between micro-service vs a "distributed monolith" (High: ↑); OpenAPI (née Swagger) (Med); JSON Schema (High: ↑); gRPC (Low: ↑); 12-factor design (High: ↑); Ent (Med: ↑).

- **Containers and Orchestration:** Docker (High: ↑), Amazon ECS (High: ↓); Kubernetes (Low: ↑).

- **Enterprise Services:** Artifactory (Expert: ↑), Jira (High: ↑), Confluence (High: ↑), GitHub Enterprise (High: ↑), GitHub (High: ↑), Pingdom (Med: ↓), New Relic (Med: ↑), Datadog (Med: ↓), Papertrail (Med: ↓), Slack (High: ↑), PagerDuty (High: ↑).

- **Databases & Key-Value/Document stores:** MySQL (Med: ↑), Redis (High: ↓), PostgreSQL (Med: ↑), Memcache (Low: ↓), Atlas (Low: ↑).

- **Metrics, Traces, and Logs:** OpenTelemetry (Med: ↑), New Relic (Med: ↑), Datadog (Med: ↓), Jaeger (Low: ↑).

- **Metadata and Config Formats:** RDFa, Dublin Core, FOAF, OpenSearch, JSON-LD, Microformats, RSS, Atom (RFC 4287), JSON, YAML, TOML, XML, HCL, Schema.org, Open Graph.

Work Experience & Notable Projects

Northwood Labs — Colorado

Owner (January 2024—Present)

Northwood Labs is an incubator for security and reliability tooling.

It is a tiny company based in Colorado who thinks that software engineering, site reliability, operations, and cybersecurity are all parts of the same whole. We want to empower teams to build quality software and reliable services, teach where there are knowledge gaps, and make it possible for every user to have access to the best in security.

Historically, most of the tools built to address these areas have done a poor job of integrating across all relevant disciplines, and can also cost a small fortune in order to help teams ensure their products and

services are well-built, reliable, and secure.

McGraw Hill (née McGraw-Hill Education) — Remote (since COVID), previously Seattle, WA

Enterprise Architect, Cloud Center of Excellence (January 2024—Present)

Moved into a position that touched the technical direction for the entire organization. Worked closely with the other members of the *Cloud Center of Excellence*, Reliability Engineering, Cybersecurity, Networking, and AppDev Engineers to ensure that we avoided "Ivory Tower Syndrome" and focused on real-world, actionable feedback and direction.

- **Documentation and Training:** Continued to ensure that people could continue to learn about new topics without requiring any specific human to become a bottleneck.
- **AWS Organizations and Control Tower:** Continued to be involved in the oversight and direction of our AWS stack, security, guardrails, and more.
- **Cross-Cloud Collaboration:** Began working alongside my peers focusing on Microsoft Azure and Oracle Cloud Infrastructure clouds. Developed my understanding of our cloud fabric which allowed high-performance networking across clouds. Worked to identify opportunities for bringing the security and guardrails that we'd developed for AWS to the other clouds, as well as train my peers on effective use of Terraform for cloud management.
- **Shifting Cloud Practices Left:** Began an ambitious, multi-year project with a high-level intention of giving us cross-cloud visibility, reporting and scoring of our cloud accounts against a list of best practices, making that data visible to everyone in the company — namely engineers performing development, and pushing that data left with the goal of getting it into developer's IDEs. This would help prevent bad practices from being used in the first place, help engineers identify where bad practices have been used in the past, and educate developers on how to build more secure and operationally-excellent software. Allows us to also view trends and changes over time to understand if we're getting better or worse across our hundreds of cloud accounts.

Principal Cloud and Platform Engineer (June 2020—January 2024)

Continuing the work I led as an engineering manager, I migrated into a more strategic role around the projects where I had started as the creator, initiator, primary developer — planning the path of the products and how they wove into the larger tapestry of our highly-heterogenous application ecosystem which had grown by way of acquisition over the years. With no longer having direct reports, I was able to focus on *technical leadership* without the responsibility of *human management*.

- **Documentation:** Prolific documentarian. Documentation is worth 50% of your grade.
- **Reliability Platform:** Products that I had personally pioneered (ECS-optimized Base AMI, Prism, Monitoring-as-Code, Terraform modules) became core pieces of our "reliability platform" alongside off-the-shelf software/services such as AWS Control Tower, Artifactory, GitHub Enterprise, GitHub

Actions, Circle CI Enterprise, Jenkins, and more.

- **Control Tower:** Partnered with McGraw Hill Enterprise Architecture and AWS Professional Services to deploy AWS Control Tower and AWS Identity Center. Lowered costs and increased control over account guardrails. Enabled automated provisioning of new accounts, and developed smoke tests as a post-provisioning validation step.

- **Clarity in Complexity:** Collaborated on the Guardrails (mandatory + custom) deployed across all AWS account *organizational units* (OUs). These were written as CloudFormation YAML, Python, and Bash scripts. In such a large complex, project, it's easy for the code to become obtuse and difficult to trace. Worked with my team to make sure we understood the fine details of the implementation, then implemented Lambda functions and CI code to read certain changes in Git commits to master/main and generate README/Confluence documentation with directed graphs and charts generated from DOT documents, to make the workflows and details easier to understand visually.

- **Base AMI program** Took what we'd learned about Packer, CIS Benchmarks, security patching, and the needs of a particular AMI's audience to develop a single build pipeline which brought the best ideas together — automatic dev builds with unit/integration testing on Git commit, production builds with complete package indexing on Git tag, pre-installing and pre-configuring agents for metrics and cybersecurity, automated security analysis scanning, making the Base AMIs available to all ±200 AWS accounts, rotating the hosts to use the new AMI with zero downtime. Adopted EC2 ImageBuilder and automated AMI rotations in the process.

- **Streamlining:** Combined elements of Terraform, Monitoring-as-Code, Base AMIs, and our custom security tooling to empower application teams to bring a Docker image with a small amount of configuration and deploy it to one of our Amazon ECS clusters with best practices, infrastructure monitoring, and operational tooling built-in, lowering overall costs.

- **Preventative automation:** Scanned Route 53 and other DNS providers to obtain a mapping of our thousands of active websites. Leveraged highly-concurrent, scalable bots to fetch certificate data from each endpoint. Enabled faster rotation for expiring datacenter certs by knowing both WHICH certs and WHERE they were installed. Verified the required DNS records for self-rotating *Amazon Certificate Manager* certs.

- **Prism:** Developed custom security and operational tooling where off-the-shelf tools wouldn't give us what we needed. Solution involved highly concurrent and dynamically-scalable nodes that would scan the AWS APIs to understand the current posture of ±200 AWS accounts. Made the data transparent to ALL engineers, enabling teams to be involved in improving their infrastructure stacks.

- **Self-hosted GitHub Actions runners:** Team adopted Amazon EKS and `summerwind/actions-runner` to deploy self-hosted runners for GitHub Actions in our GitHub Enterprise environment. Wrote smoke tests which ran every hour to validate the GitHub Actions runner environment, as well as the actions we'd imported into GitHub Enterprise for internal developers to use. This provided both increased visibility as well as a working example of how to leverage the actions effectively.

- **Automation for Artifactory:** Rebuilt our Artifactory cluster with a "cattle, not pets" approach. Dedicated Base AMI, rotated monthly. Migrated artifacts from NFS to S3. Rewrote configuration in

Terraform instead of by-hand. Moved service-user management into Terraform. This automation reduced the amount of human error in the process, improved our security posture, and increased consistency leading to a better developer experience.

- **Custom Packages:** Worked to streamline the developer experience by moving all disparate Amazon ECR Docker image repositories into Artifactory. Worked to reduce the time to build VMs and Docker images by identifying the common software people were manually installing, and began packaging them as pre-compiled `.rpm`, `.deb`, and `.apk` (Alpine Linux) packages that could be installed from Artifactory through the system's built-in package management system. Faster builds with better reliability and reduction of the "left-pad" problem.

- **Token Vending Machine:** Built a Token Vending Machine to enable continuous token/password rotation for our engineering teams. "Push button, receive token." Solution leveraged Secrets Manager, Lambda, KMS, IAM policies, and some custom CLI software written in Go. First integration was for service-users (robots) in Artifactory.

- **Training and Education:** Worked to develop the SysAdmin "button pushers" on my teams into more well-rounded software engineers who could automate more reliabily. Continued to push to *raise the bar* in the quality of our team. When SysAdmins left the company, worked to hire *true* SREs to fill their spots.

- **ARM64 Adoption:** When Apple announced ARM64-based *Apple silicon* Macs in November 2020, it became obvious to me that ARM64 was going to play a large role in our future. I began chipping-away at the places where Intel `x86_64` was *assumed*, updating our custom package build pipeline for ARM64, adding ARM64 runners for GitHub Actions, adding ARM64 parity in Artifactory for the remote repositories we were proxying, writing tutorials and hands-on documentation for using Docker BuildKit to produce multi-platform container images, and more. When the company began seriously considering AWS Graviton (ARM64) CPUs for cost reasons, all of the pieces for adoption were already in-place. This also addressed issues faced by multi-platform development teams (Intel Macs + Apple silicon Macs).

Engineering Manager, Site Reliability (October 2018—June 2020)

Owned, and was the key decision-maker for the development of a core platform of company-wide, reliability-oriented projects. With our development teams moving toward Full-Cycle Development, our SRE team focused on solving more macro-oriented problems which affected more than 75 decentralized, heterogenous engineering teams across the company. These projects have empowered greater self-service for engineering teams, enabling them to move faster without having to reinvent the wheel.

Many of the following projects got their start in my work as an application engineer for McGraw Hill, and carried over into this role.

- **ECS-optimized Amazon Linux Base AMI** for all Amazon ECS applications. Modified the version vended by AWS to meet Level-2 CIS Guidelines for both Amazon Linux and Docker. Underwent deep collaboration with security, operations, and various business units to ensure compliance. Achieved

high levels of opt-in adoption, which gave security and operations orgs higher levels of confidence in the product development teams.

- **Prism** which is an "executive dashboard" enabling significantly improved visibility into the security and operational configurations of our AWS accounts (hundreds). Enables visibility to Engineering Managers, Directors, VPs, and the CTO, while also providing clear instructions to app engineers to understand why the configuration is incorrect and what needs to be done to resolve the issue.

- **Monitoring-as-Code** which leverages Terraform and Python to streamline the process of generating and maintaining dashboards and monitors in Datadog and New Relic across a large, heterogeneous swath of applications. Trained development teams in adopting "full-cycle" development practices where the development team owns day-to-day operations of their services including deployments, support, and on-call rotations.

- Formed a leadership group to develop a more rigorous process for developing, patching, vending, and maintaining re-usable **Terraform modules** that are used by large numbers of product development teams across the company. Standardized their development, contribution, and usage guidelines, adopted an Apache-style “incubator” for developing new modules, and adopted a process for shipping LTS-style packages of modules.

- Took over engineering management responsibilities for the **Site Reliability** group in MHE’s Seattle office. Worked to integrate our office better with the larger, developing SRE practice across all offices across the U.S. Joined the SRE leadership group to help guide and participate in the development of better processes around reliability, which we then worked with product development teams to adopt and apply.

- Rebooted our Seattle SRE **interview process**, with a much higher focus on identifying high-quality engineers with a 70/30 split between software engineering (Dev) and systems engineering (Ops), and who were more *leaders* than not. Integrated many ideas and *leadership principles* from my time working at AWS.

- Adopted a more integrated, SRE-style of working alongside development teams, and (mostly) ended the practice of dev teams “tossing things over the fence” to some Ops team in the parts of the org that the Seattle SRE team supported.

Staff Software Engineer (October 2016—October 2018)

Ryan led the development of multiple tier-1 services as part of the educational content authoring pipeline, leveraging REST, GraphQL, API design, AWS, Amazon ECS, Docker, Terraform, ePubs, and security best practices. Led the technical direction of the projects, socialized them, documented them, and provided ongoing guidance around their design and use.

- Lead the development of the authoring component of McGraw Hill’s SmartBook 2.0 product, and the internal system which indexes authored content, builds ePubs, and encodes images/video for McGraw Hill’s ePub CDN.

- Kickstarted the use of continuous integration, continuous delivery, rapid deployments, Docker containers, and “dog-fooding” newer processes which allowed deployments that were both more frequent and more reliable.

- Introduced a more hands-on monitoring style, enabling dev teams to be more actively engaged in their own operations instead of relying exclusively on an external, third-party vendor used by other groups in the company. Empowered significantly-lower MTTR during incidents, tracking app-level metrics, and the introduction of KPIs.

- A member of the core team that was migrating all new infrastructure to "Infrastructure-as-Code" tooling such as Terraform, Packer, etc. Identified patterns across applications, and began the effort to streamline infrastructure maintenance with shared, re-usable Terraform modules.

Perimeter of Wisdom, LLC

Co-Owner, CTO, Producer (February 2015—2018)

On the technical side, Ryan built the entire “The First-Time Offender’s Guide to Freedom” website, soup to nuts. Ryan also performed all of the production work on the eBook, authored by E. M. Baird.

- Leveraged modern tools to build the front-end, including Bootstrap, LESS, JavaScript, Gulp.js, npm, Bower. Ryan built the back-end in PHP 5.6, using HHVM and Nginx, MySQL, Redis, Slim Framework, Monolog, Pimple, Twig, Guzzle, Doctrine, Phinx, and Symfony components. Ryan deployed the application using Ansible, and developed the application in a Vagrant environment running Ubuntu.

- Runs the unit, integration and functional tests using PHPUnit, Behat, Mink, and Selenium. Ryan leverages Amazon SES for sending email, Amazon S3 for static file storage, Stripe for payment processing, Linode for web hosting, MaxMind IP-based geolocation, and Google Books and Dropbox for ensuring that customers always have the latest errata fixes.

WePay — Redwood City, CA

DevOps Engineer (April 2015—September 2016)

- Improved how WePay provisioned cloud infrastructure, deployed updates, managed security patches, monitored applications and infrastructure, and streamlined the process of planning, developing, deploying and maintaining new micro-services throughout the company.

- Led the cross-company effort to upgrade the monolithic application’s software stack from PHP 5.4 to PHP 5.6. This required cross-team collaboration across all of the major engineering teams, QA, and replacing over 200 servers across multiple environments with zero customer-facing downtime.

- Maintainer of multiple tier-1 systems including Artifactory, GitHub Enterprise, Toran Proxy and Phabricator.

Senior API Engineer (April 2014—April 2015)

- Developed new API endpoints to help expand WePay's business and support its partners.
- Was instrumental in designing/developing WePay's MFA-as-a-Service offering ("System and Methods for User Authentication across Multiple Domains" (US15042104; Pending)).
- Heavily involved in the security of WePay's products, coordinating fixes with teams against other priorities, and fixing the issues himself in many cases.

Truncated

Earlier experience is available upon request.

Groups & Accomplishments

- Editor, Producer, and Publisher for the book *Federal Probation Bible, 2022–2023 Edition* written by E.M. Baird. (ISBN: `978-0-578-99269-3` (Paperback))
- Voting Representative for AWS, PHP Framework Interoperability Group (2012–2013)
- Member, RSS Advisory Board (2007—2009)
- Patent, "Hive-based Peer-to-Peer Network" (US8103870B2)
- Patent, "System and Methods for User Authentication across Multiple Domains" (US15042104; Pending)
- Student guest speaker for the 2004 Silicon Valley College graduation ceremony.

Education

Carrington College California (née Silicon Valley College) — San Jose, CA

Bachelor of Arts, Design and Visualization (November 2003)

- GPA: 3.84
- Web, graphic, multimedia, and publication design.