CHEF HABITAT™

Application Delivery Automation

Chef Habitat is an application delivery automation solution to help teams eliminate run-time errors, minimize release delays and accelerate software delivery to any environment. Application requirements are defined with the application during development in order to shift-left failure identification from run-time to build-time. Application delivery teams can rest assured that what is built in development will behave consistently in any environment - containers, VMs, bare-metal, etc.

Benefits

- **Productivity:** eliminate thousands of hours spent configuring, validating and remediating builds.
- **Consistency:** build applications using tested configurations with defined dependencies.
- **Agility:** deploy builds on-demand to any environment without any refactoring or rewriting.
- **Quality:** eliminate run-time errors and expensive failed deployments.
- **Visibility and validation:** view and validate delivery status of packages in real-time across all environments.

Features

- **Reusable content:** hundreds of pre-defined configuration templates and a robust open source user community.
- **Modular:** each dependency is built as an independent versioned plan.
- **Codified instructions:** automation stored as coded artifacts that are easy to search and share.
- **Autonomous supervisors:** lightweight agents that automate installation, updates, monitoring, and provide APIs for reporting.
- **Immutable objects:** what is built and run in development will be exactly the same in production.
- **DevOps integration:** easy to integrate with Jenkins, Azure DevOps, Terraform and others.
- **Day 2 support:** health and compliance checks included during the build phase.

Challenges

In order to survive in the new digital era, organizations must be able to adopt and deliver new technologies and features faster than their competitors. As a result, most organizations have adopted a multi-modal approach to IT delivery. New applications are built using agile and cloud-native approaches. Older applications are scheduled for refactoring. The responsibility for different components falls to different teams that are siloed and use different delivery methodologies.

This has resulted in repetitive work being done across teams, inconsistent build processes, tool-sprawl and an increase in the number of failed deployments. Things that work in dev don't work in pre-prod, things that worked in pre-prod fail in production - all of which result in expensive rework and delayed releases. Savvy organizations are now realizing that in order to drive innovation across their organizations they must better manage application requirements and enable delivery consistency across all of their applications.

Solution

Chef Habitat automates the process of defining and packaging codified configuration artifacts that are stored and versioned along-side an application binary. It doesn't matter to Habitat how an application was developed or where it will be deployed. The process is the same. The application is abstracted from the underlying operating system, then the run-time dependencies (like .NET or JRE) and build-time dependencies (like the jdk or gcc) are defined, along with instructions on how to initialize the app. Codified build packages can be easily integrated with other DevOps solutions like Terraform and Jenkins.

Chef Habitat uses patented technology to report on the status of an application so teams get instant validation when an application has been successfully deployed. Supervisors (intelligent, light-weight dynamic agents) are used to auto-update applications when underlying dependencies change. Ongoing support and compliance are no longer an after-thought. Management and audit capabilities are included in the build package and travel along with the application wherever it is being deployed. All of this helps organizations ensure that any initial savings gained from the implementation of a new technology aren't outweighed by higher operational overhead and compliance remediation costs in the future.

Common use cases

- Agile Development
- Application Operations
- Cloud Migrations
- Continuous Delivery
- Edge Computing
- COTS and Middleware Delivery
- DevOps Organizational Alignment
- Microservices / Cloud-Native Applications
- Modernizing Legacy Applications
- Software Engineering

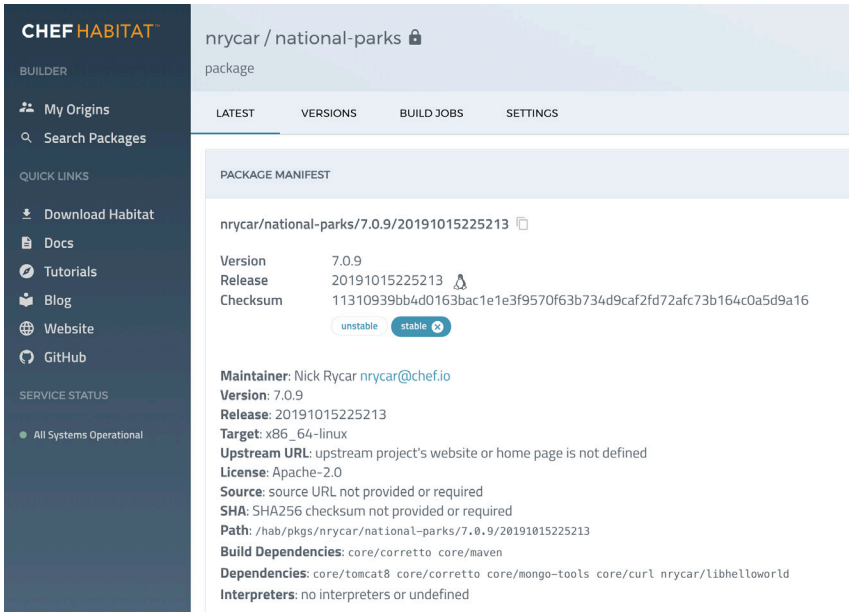
Why Chef Habitat?

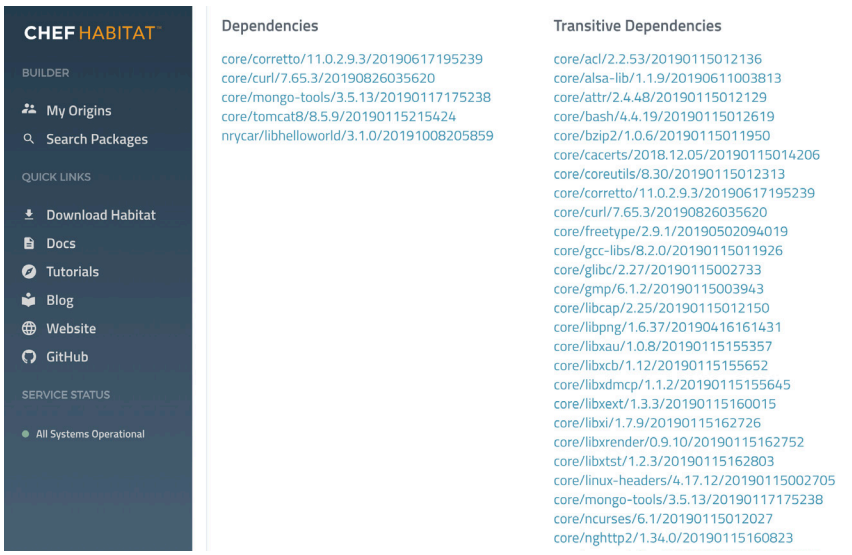
Chef Habitat is the only automation tool of its kind that enables companies to apply a consistent approach to application definition, packaging and delivery across all applications and environments. For organizations looking to safe-guard investments in DevOps tooling and scale continuous delivery across all applications Chef Habitat is the clear choice.

A shift-left approach to application delivery

Chef Habitat improves the way teams work together to build applications. A Habitat plan defines the dependencies required to build and run each application. Each dependency has a plan of its own, maintained by its respective owner. Plans are stored in a single repository where they can be easily searched, shared, updated, customized, and versioned.

Once a plan is defined, the magic of Chef Habitat takes over. When an application is built, the resulting artifact contains metadata pointers for its full dependency chain. This ensures that the artifact a developer tests on their laptop remains consistent with the ones running in production, regardless of provider or platform.





Build once. Run anywhere.

Chef Habitat packages can be exported to run in a variety of runtimes with zero refactoring or rewriting. Current supported export formats include:

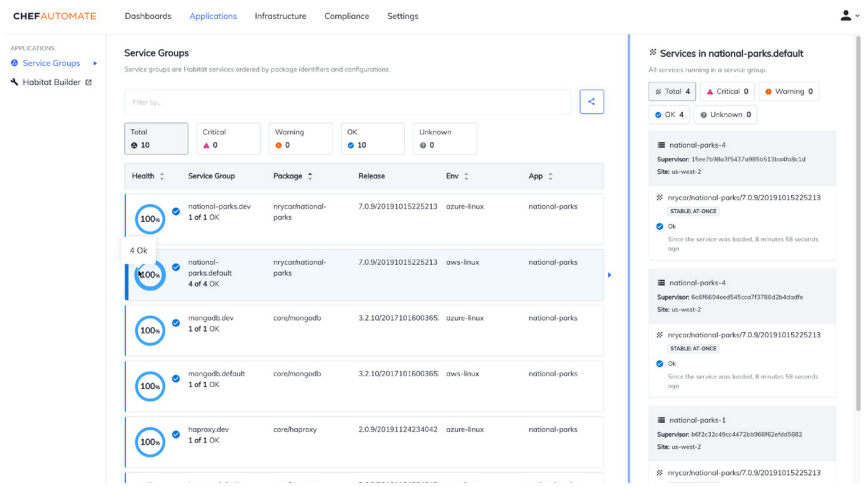
- Docker
- Tarball
- Application Container Image (ACI)
- Apache Mesos an DC/OS
- Cloud Foundry (Docker)
- Cloud Foundry (Endpoint)

Minimized and highly portable build artifacts

Chef Habitat builds optimized packages that only contain what the application needs and are constructed independently of any underlying infrastructure. Habitat does not care what technology was used to construct an application, nor does it care about the destination. Whether you want to run an application in a cloud-native environment, on bare metal, in the cloud, or in a container, you can. This not only enables organizations to accelerate adoption of technologies like Kubernetes but also minimizes the time and effort needed to create and version build packages to support multiple environments.

Real-time visibility and validation across all environments

The Application Dashboard in Chef Automate allows teams to organize and display data from Chef Habitat in an intuitive way. With a click of the mouse, users can see what applications have been updated, what versions are running where, what the application status is and where there are failures. Chef Automate also provides teams with easy access to audit and configuration data for infrastructure and cloud assets.



Related products/solutions

CHEF AUTOMATE™

Enterprise dashboard and analytics tool enabling cross-team collaboration with actionable insights for configuration and compliance and an auditable history of changes to environments.

CHEF INSPEC™

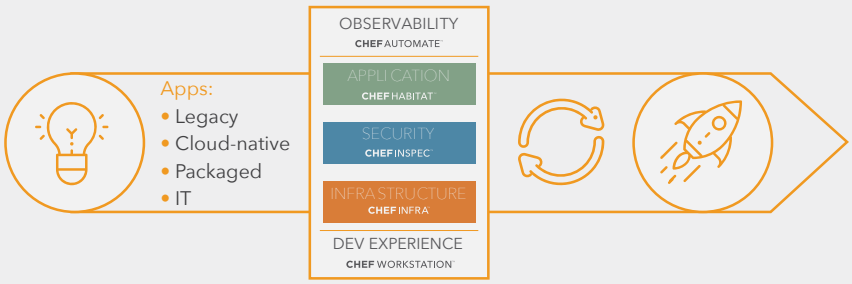
Define and continuously enforce security and compliance standards on-prem and in the cloud.

CHEF INFRA™

Configure and remediate any number of systems across desktop, cloud and private data centers.

Chef Enterprise Automation Stack™

A comprehensive automation platform for DevOps and security teams to build, deploy, manage, and secure any application running on any infrastructure:



Explore Chef Habitat today!

Chef Habitat documentation: https://www.habitat.sh/docs

Learn Chef Rally free online learning: https://learn.chef.io

CHEF

Seattle, WA | PH: 206.508.4799 | sales@chef.io | www.chef.io