

Chef Infra Client 18

Overview of New Features and Best Practices

Updated October 2022

PRODUCT GUIDE

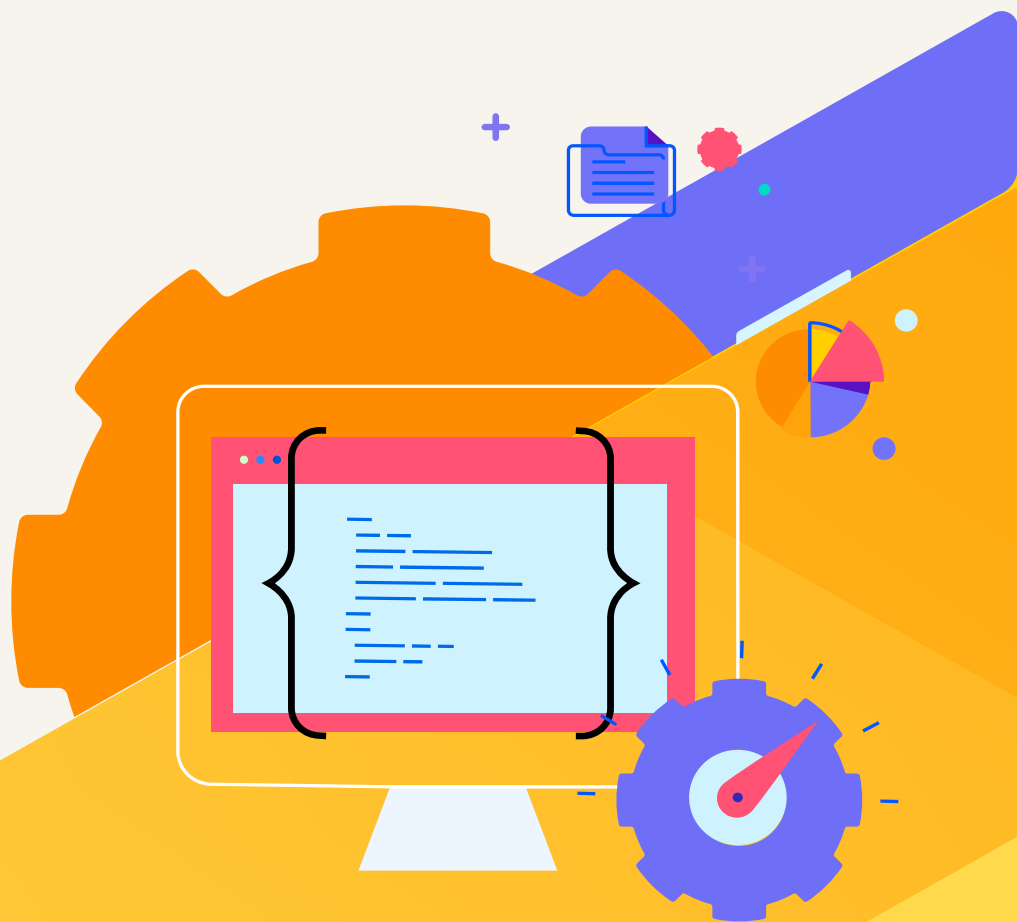


Table of Contents

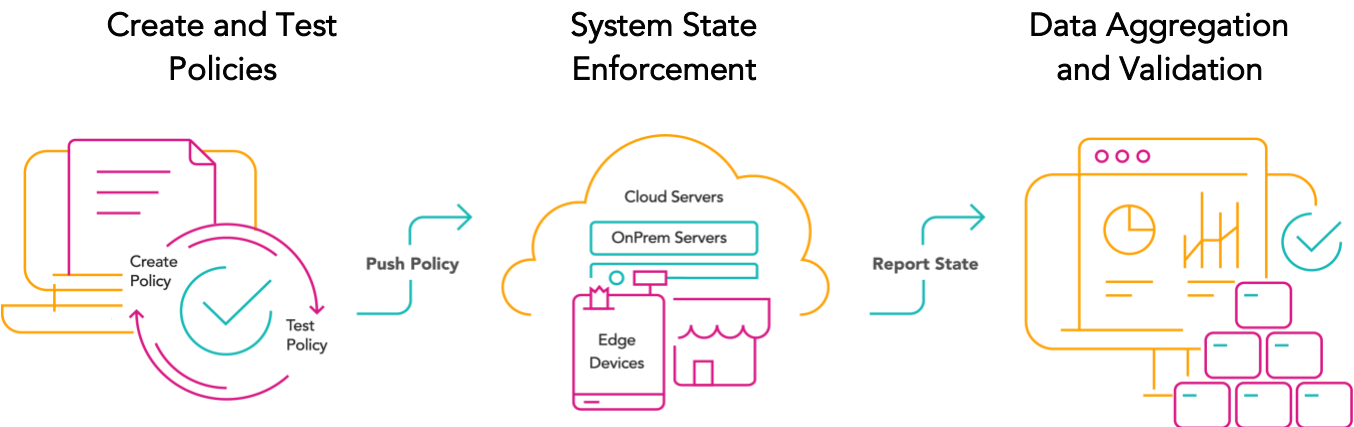
Overview.....	3
<i>Chef Infra Client 18 Key New Features</i>	<i>7</i>
Test Driven Development (TDD).....	9
Embedded Profiles	10
System Coverage Enhancements	11
Expanded Cloud Support.....	12
User Experience Improvements.....	12
Full-Stack Visibility	14
Windows Patching	16
Support and Security Updates.....	17
<i>Chef Infra Client Release History</i>	<i>17</i>
Chef Infra Client 18.....	17
Chef Infra Client 17 (to 17.5)	18
Chef Infra Client 16.....	21
Chef Infra Client 15.....	22
Chef Infra Client 14.....	23
Chef Infra Client 13.....	24
Chef Infra Client 12.....	25
<i>Upgrade Preparation Useful Info</i>	<i>26</i>
Chef Infra Server Versions	26
Chef Workstation and ChefDK.....	26
Upgrading Cookbooks & Clients	26
<i>Upgrading Cookbook Guidance and FAQ.....</i>	<i>27</i>
Chef Upgrade Lab	27
Which offenses do I fix first?.....	28
Will correcting these offenses break my current chef-client runs and/or pipeline build runners?	31
Can I upgrade from Chef Infra Client 12 to 18?	31
How should I release these changes in a controlled fashion?	31
How can I handle this in the future?.....	33
<i>Upgrading Chef Infra Client Guidance.....</i>	<i>34</i>
The Manual Path.....	34
The Accelerator chef_client_updater	34
<i>Additional Resources.....</i>	<i>36</i>

Overview

Chef Infra is a configuration management tool for defining infrastructure as code (IaC), making the process of managing configurations and system state automated and testable. Unlike other configuration solutions, Chef Infra takes a policy-based approach that builds upon the principles of test-driven development and idempotency. System configuration and application-change testing can be done in parallel, so system states are easily enforced and monitored across all infrastructure and teams. With Chef Infra, users define configurations once and then apply them across mixed fleets of Linux, macOS and Windows systems, regardless of OS version and architecture. When paired with Chef InSpec for compliance, Chef Infra enables true Policy as Code to make your all systems are up to date and secure.

Chef Policy-Based Infrastructure Automation Architecture

Using Chef to automate configuration management allows DevOps teams to define policies that are repeatable, consistent, and reusable. The result is increased business agility and security because all systems and resources are continuously and automatically evaluated, corrected, and modified.



Chef Workstation	Chef Infra Server and Client	Chef Automate
<p>Reduce risks by iterating on policy changes before pushing them to production.</p> <ul style="list-style-type: none">• Chef Tools: Chef Infra Client, Chef InSpec, Chef Habitat, Cookstyle, Test Kitchen, Chef CLI and knife• Chef Language: Pre-built resources for managing systems and helpers that make authoring and distributing cookbooks easy	<p>Enforce policy by converging the system to the state declared by the various resources.</p> <ul style="list-style-type: none">• Planned, unstructured and policy-based updates• Dynamic behavior support• Ephemeral resource management• Secure approach and single agent for configuration and compliance	<p>View and validate intended and actual state across all systems.</p> <ul style="list-style-type: none">• System Policy management• Real-time interactive dashboards• Role-based access controls• Third-party integrations• Data APIs

Note on Chef Infra Server: Chef Infra Server does not follow the same yearly cadence as Chef Infra Client. The latest release is Chef Server is Chef Server 14.x, which is compatible with all recent Chef Infra Client releases. For more about Chef Server 14 read the blog “Product Announcement: Chef Infra Server 14 Now Available”.

Over the past 10 years a lot has changed within the Chef Infra product portfolio. We’ve continued to evolve our solutions, making them easier to integrate and use, adding new

functionality and increasing scalability. The table below provides a summary of key changes made within the portfolio in the last 5 years.

Table 1: Chef Infra Past and Present: Summary of Key Platform Changes Made in the Last 5 Years

Chef Infra Past		Chef Infra Present
Ruby	Language	Chef Language (Ruby-based), resources, and helpers
ChefDK, ChefSpec, and Foodcritic	Developer Kit	Chef Workstation, Test Kitchen, and Cookstyle
Roles, Environments, and Audit Cookbook	Content and Compliance	Policyfiles, InSpec Profiles and Chef Infra Client Compliance Phase
Stand Alone Chef Server and Chef Manage	Management	Chef Automate with Integrated Chef Infra Server
Adhoc Testing	Testing	Test Kitchen, Chef CLI, Chef InSpec and Test-Driven Development Adoption
Manually Pushing Changes	Deployment	CI/CD Automated Pipelines, GitHub Actions, and Pull Requests

Key product notes include:

Chef Language, Resources, and Helpers: The [Chef Infra Language](#) is a comprehensive systems configuration language with resources and helpers for configuring operating systems. The Chef Infra Language is based on Ruby, allowing you to utilize the power of Ruby when the built-in language doesn't meet your needs out of the box. With the Chef Language, users define configurations once and apply them across mixed fleets of Linux, macOS and Windows systems, regardless of OS version and architecture.

- **Resources:** Resources are used for configuring components such as packages, files, directories, or firewalls. Today, Chef Infra Client ships with more than 165+ resources for common automation tasks such as user, file, kernel_module and windows_task.
- **Helpers:** Helpers enable users to make configuration decisions based on operating systems, clouds, virtualization hypervisors, and more.

ChefDK End of Life: ChefDK was first shipped in 2014 and on **Dec. 31, 2020, reached end-of-life status**. Chef Workstation launched in 2018 has since been replaced ChefDK. Chef Workstation includes all the tools you're familiar with in ChefDK, plus much more. We've packaged new tools for working with VMware, executing ad-hoc jobs, updating your cookbooks, integrating InSpec compliance, and bootstrapping system. For more information read the blog ["Goodbye ChefDK, Hello Chef Workstation"](#).

Chef Cookstyle: Cookstyle is a code-analysis and linting tool built upon [RuboCop](#) that replaced Foodcritic in September 2019, and ships as part of Chef Workstation. Today, Chef Cookstyle includes nearly 250 Chef Infra specific rules (called *cops*) that catch common cookbook coding mistakes, clean up portions of code that are no longer necessary and detect deprecations that prevent cookbooks from running on the latest releases of Chef Infra Client.

Policyfiles: Policyfiles are the best way to handle dependencies and change management across your Chef Infra managed infrastructure. They combine the very best parts of Roles, Environments, and Berkshelf into a single workflow. A single *chef install* command bundles multiple cookbooks, profiles and dependencies into an immutable object. Immutable Chef Infra Policyfiles cannot be changed once bundled, ensuring your configurations change only when you want them to. The Chef Infra Client no longer recalculates dependencies at the start of every run, making them faster and more efficient. For more information view the [Policyfile documentation](#).

Test Kitchen: Test Kitchen is an independent open-source project sponsored by Chef that provides real-world test environments in containers, VMs and cloud instances to execute infrastructure code on one or more platforms in isolation. Test Kitchen is Chef's integration testing tool of choice for cookbooks. It's included as part of Chef Workstation and used by all Chef-managed community cookbooks.

Chef Infra Client 18 Key New Features

The release of Chef Infra Client 18 includes a number of new features and improvements, including a renewed focus on Test Driven Development (TDD), enhanced system coverage for different architectures and cloud instances, and more tools that make using and upgrading Chef Infra easier.

Key new features and enhancement introduced in Chef Infra Client 18 include:

- Slow Resource Reporting with `--slow-report` flag
- Ability to allow or deny attributes that are reported to Automate
- Secrets Management Beta
- Run Lists can now be specified when using Policyfiles
- Chef Solo recipe fetching for S3 using `--recipe-url s3://BUCKET/file.tar.gz`
- Improved VMware detection
- Updates to Windows, macOS resources
- Improved Compliance Phase integration between Chef Infra and Chef InSpec

Resource updates and improvements:

- Windows updates
 - `windows_firewall_rule`: multiple IP support
 - `windows_pagefile`: entirely rewritten
 - `windows_printer`: entirely rewritten
 - `windows_printer_port`: entirely rewritten
 - `windows_security_policy`: Improvements to value handling
 - `hostname`: Entirely rewritten on Windows
 - `powershell_package`: support for passing an array of install options and performance improvements
 - `windows_feature`: Performance improvements
 - `chef_client_scheduled_task`: support for setting task priority
 - `registry_key`: support for lazying properties
 - `powershell_package_source`: support for authenticated sources
 - `chocolatey_source`: support for authenticated sources
- macOS updates
 - `group`: support for setting GID as integer
 - `homebrew_cask`: support for `-` or `@` in the names
 - `user`: handles missing fields in existing users better
 - `macos_userdefaults`: Entirely rewritten with native API calls
 - `chef_client_launchd`: Handle restarts when the config changes

- Improved management of RHSM on RHEL
- New Resources
 - habitat_package, habitat_sup, habitat_config, habitat _install, habitat_service, habitat_user_toml
 - windows_defender / windows_defender_exclusion
 - windows_update_settings
- Compliance Phase
 - Improved logging
 - JSON-file reporter *off* by default
 - Infra attributes passed to InSpec by default
 - Embedded InSpec profiles in cookbooks
 - Interval run support
- InSpec improvements:
 - 29 improved resources
 - 16 new resources
 - windows_firewall
 - windows_firewall_rule
 - selinux
 - zfs_pool
 - zfs_dataset
 - mongodb_conf
 - mongodb_session
 - opa_cli
 - opa_api
 - cassandradb_conf
 - cassandradb_session
 - mssql_sys_conf
 - oracledb_conf
 - oracledb_listener_conf
 - ibmdb2_conf
 - ibmdb2_session
- Packaging
 - Arm64 containers published for testing
 - FIPS PPC RHEL support
 - Solaris 11.3 support
 - macOS 12 support
 - Windows 11 / 2022 support
 - The return of Ubuntu 16.04
 - Debian 11 support

- Support for AlamaLinux and Rocky Linux
- Infra Language
 - New render_json, render_toml, and render_yaml Infra Language helpers

For a full list of features & enhancement visit docs.chef.io.

Test Driven Development (TDD)

Improvements to Test Kitchen build on updates made in the release of Chef Infra Client 18, and make it easier than ever to test cookbooks and policies using Docker, Vagrant or cloud environments. Users can spin up multiple environments and OSes at once and add suites to include cookbooks and policies to run automatically. We've created Docker (Dokken) images that simulate true OS environments so you can test with confidence.

With improvements to Compliance Phase, users also can use their integrated profiles in Test Kitchen. For example, in the kitchen.yml file under *suites*:

suites:

- name: default
 - verifier:
 - inspec_tests:
 - compliance/profiles/<profile-name>

```

1  ---
2  driver:
3    name: dokken
4    use_sudo: false
5    privileged: true # allows systemd services to start
6
7  provisioner:
8    name: dokken
9
10 transport:
11   name: dokken
12
13 verifier:
14   name: inspec
15
16 platforms:
17   - name: ubuntu-22.04
18     driver:
19       image: dokken/ubuntu-22.04
20       pid_one_command: /bin/systemd
21   - name: centos-8
22     driver:
23       image: dokken/centos-8
24
25 suites:
26   - name: default
27     verifier:
28       inspec_tests:
29         - compliance/policies/myproject
30
```

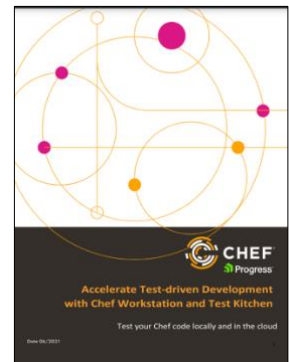
To learn more about testing with Docker read the blog and watch the on-demand webinar - [Chef Infra Best Practices: #3 Testing Chef Infra Cookbooks Fast with Docker](#).

Starting with Chef Infra Client 17 and now in version 18, a number of Policyfiles improvements were made to make them easier to migrate to and bridge the gap between traditional Berks workflows and Policyfiles. Improvements include:

- Users can now optionally execute Chef Infra Client with a specified run list on nodes that are managed with Policyfiles.
- Run lists with Policyfiles give users the safety of locked sets of cookbook dependencies while also giving users the flexibility to change run lists or run different run lists on nodes for ad hoc Chef Infra Client converges.
- Policyfiles with run lists offer additional flexibility over named run lists and are better suited for ad hoc Chef Infra Client execution or programmatically changing run lists during bootstrap.

To get started with Chef Test Kitchen check out the blog and New User Guide:

[Chef Guide: Accelerate test-driven development with Chef Workstation and Test Kitchen](#)



Embedded Profiles

New since Chef Infra Client $\geq 17.5.22$ are three new resources that make it easier to ship Chef InSpec profiles, waivers and inputs, allowing you to combine infrastructure and compliance in a single artifact. Bringing infra and compliance together ensures security is always considered when making changes to your systems and enables collaboration in DevSecOps through shared pipelines. Combining infrastructure and compliance content in cookbook artifacts also allows for the safe and controlled promotion of compliance content from development to production using Policyfiles.

A new *compliance* directory is created in your cookbook directory when using *chef generate cookbook <cookbook-name>* that includes subfolders for *inputs*, *profiles* and *waivers*:

```
my_cookbook/
├── compliance/
│   ├── inputs/
│   │   └── my_inputs.yml
│   ├── profiles/
│   │   ├── my_profiles/
│   │   │   ├── controls/
│   │   │   └── inspec.yml
│   └── waivers/
│       └── control_waiver.yml
└── metadata.rb
```

You can then reference those files using Chef notation:

Load a single profile from a specific cookbook:

```
include_profile "my_cookbook::profile_name"
```

Load all waivers from a cookbook:

```
include_waiver "my_cookbook::.*"
```

Load all inputs that start with ssh:

```
include_input "acme_cookbook::ssh.*"
```

System Coverage Enhancements

DevOps teams want fewer tools, not more, and with Chef Infra 18 we've continued to expand commercial and community support for all the platforms that make up your infrastructure fleet. Highlighted coverage enhancements in Chef Infra Client 18 include:

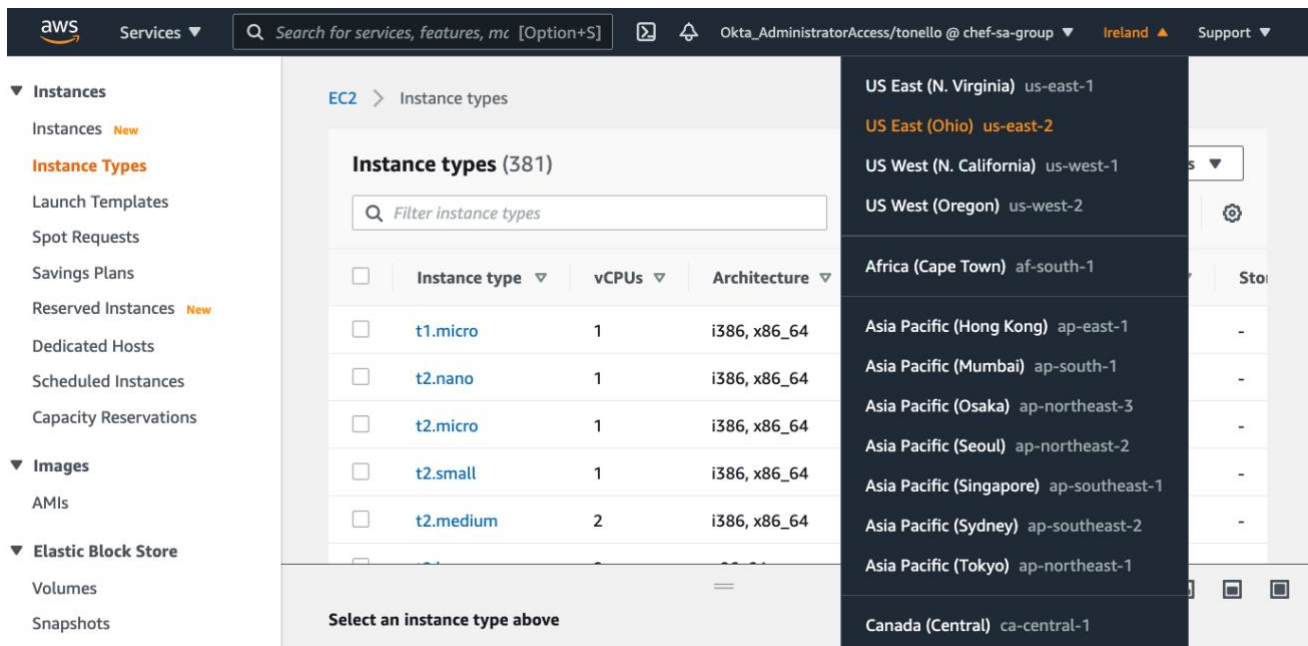
- **ARM:** Improved support for ARM platforms including better ARM detection within architecture helpers and new packages for Linux distros on ARM
- **MacOS:** Improved support for macOS profiles, Big Sur support, M1 processor support, macOS Monterey support and new Homebrew update resource
- **Windows:** Ongoing Windows support enhancements including: production of Windows 8 packages; PowerShell Core support; resources for managing Windows Update and Windows Defender; new Windows audit policy resources; and firewall profile resources
- **Linux:** Large improvement to Linux CPU detection, support for FIPS on Ubuntu and PPC RHEL, detection of new RHEL variants such as Rocky Linux and AlamaLinux, support for Podman, and improved RHEL 8 packages

Whether you're running Linux, macOS, Windows, ARM or cloud-based instances, RHEL 8, macOS 11, and others Chef Infra Client can manage them with improved support for native architectures. That means you can code once and apply everywhere. [View our full list of supported platforms.](#)

Expanded Cloud Support

Chef has long supported all the popular cloud providers and Chef Infra Client 18 includes additional improvements that let you to take full advantage of your AWS, Azure, or Alibaba environments. Now, you can use cloud variables like security groups, GEO location and region to target your Chef configuration management, helping to ensure you're properly securing and diversifying your critical workloads.

At [ChefConf '21 Online](#) we launched a Secrets Management integration for fetching secrets from AWS Secrets Manager, Azure Key Vault, HashiCorp Vault, and Akeyless Vault.



User Experience Improvements

Chef Infra Client 18 includes more improvements to Chef Infra resources and helpers. We've slimmed down the Chef Infra Client while expanding its capabilities making running commands more straightforward and intuitive. We've also made improvements to Cookstyle, which checks your code and can autocorrect common errors. Chef Infra Client 18 continues to make the upgrade process easier, and we improved Chef Infra's idempotency, and **improved performance by eight-fold** in the systemd_unit resource by streamlining requests. The Chef Infra language also includes new helpers for writing TOML, YAML, and JSON configuration data to disk.

Chef Cookstyle is a code analysis and linting tool built upon RuboCop is shipped as part of Chef Workstation. It replaced Foodcritic in September 2019. Cookstyle helps users write better Chef Infra cookbooks by detecting issues and automatically correcting cookbook code. Cookstyle enhancements released as part of Chef Infra Client 18 include:

- 29+ new cops added, bringing the total to 240+ cops
- New Cops for validating Chef InSpec profiles
- Majority of cops are now auto-correcting
- Comprehensive documentation added
- Numerous updates to Chef Cookstyle to improve overall reliability, performance, and integration with external tools

A comprehensive list of Chef Cookstyle cops can be found [here](#).

For more on Chef Cookstyle check-out:

[Blog/Recorded Webinar: Chef Infra Best Practices: #1 Using Cookstyle for Infra Client Upgrades](#)

Chef Infra Compliance Phase enables compliance and auditing reporting using the Chef InSpec engine as part of any Chef Infra Client run without the need for the audit cookbook. This simplifies the workflow needed to implement and run compliance audits, view results, and do analysis. It extends our policy-based approach to configuration, enabling a single agent than can handle the end-to-end workflow from state enforcement to data aggregation to validation. Additional features of Chef Infra Client phase include:

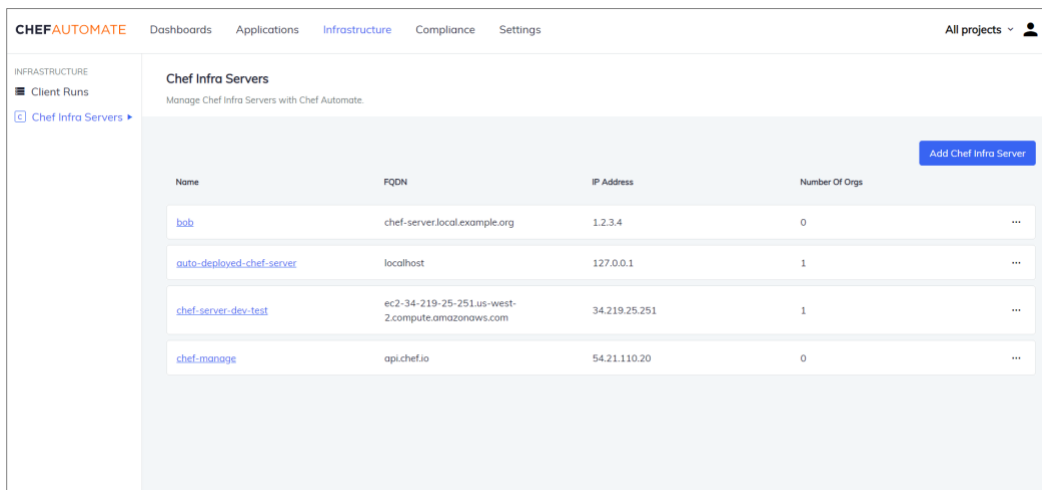
- Author and ship compliance content directly in your cookbooks combining infrastructure and compliance into a single, versioned, and immutable artifact
- Integrate Chef Infra attributes to expose infrastructure state information to compliance profiles
- Dozens of new InSpec resources to make writing compliance content easier than ever
- Improved integration to allow test-driven development for compliance, enabling DevSecOps

To learn more about Chef Infra Compliance Phase watch the [on-demand webinar](#).

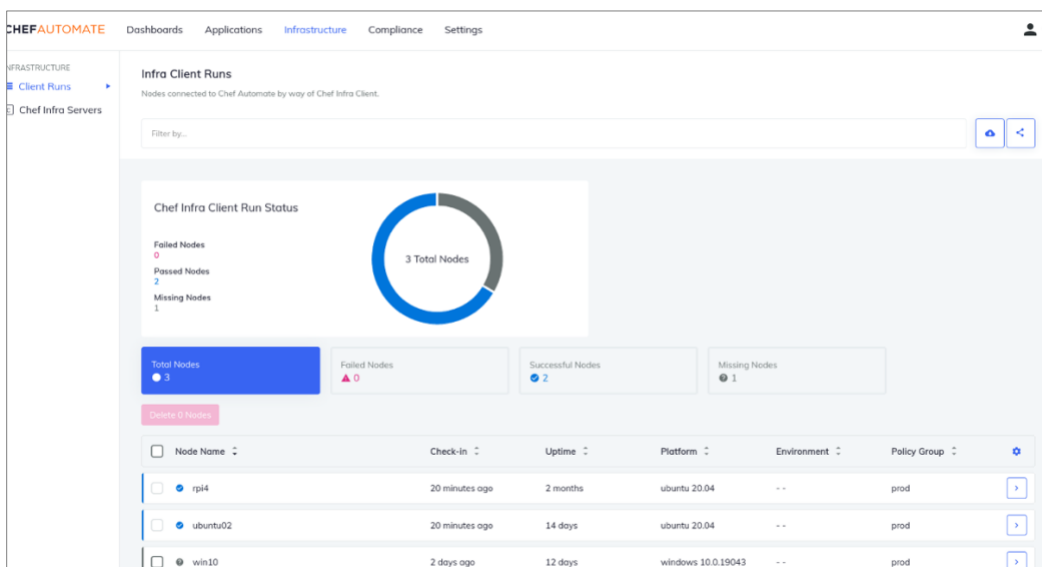
Full-Stack Visibility

The **Infrastructure State Management Dashboards** allow users to view and manage Chef Infra Server details in Chef Automate. Using these views users can:

- Add multiple Chef Infra organizations or servers to Chef Automate to review cookbooks, roles, environments, data bags, nodes, clients, and Policyfiles for each organization
- Search and find roles, environments, Policyfiles, data bag items, and nodes from Chef Automate
- Manage node reporting data back to Chef Infra Server from Chef Infra Server views in Automate
- View a history of revisions of each policyfile with associated policy group provides unmatched visibility of the given infrastructure. Combined with existing client run reports, Infra Server views give full visibility and control of underlying infrastructure.



Name	FQDN	IP Address	Number Of Orgs
bob	chef-server.local.example.org	1.2.3.4	0
auto-deployed-chef-server	localhost	127.0.0.1	1
chef-server-dey-test	ec2-34-219-25-251.us-west-2.compute.amazonaws.com	34.219.25.251	1
chef-manage	api.chef.io	54.21.110.20	0



Example: View all the cookbooks of an organization

[Chef Infra Servers](#) > [Organizations](#) > 4thcoffee

4thcoffee

Projects
(unassigned)

Cookbooks

Roles

Environments

Data Bags

Clients

Nodes

Policyfiles

Name	Cookbook Version
aix	2.4.6
apache2	8.7.0
audit	9.5.1
centos-cookbook-file	0.1.0

[Chef Infra Servers](#) > [Organizations](#) > [Roles](#) > chef-load-role-544188300

chef-load-role-544188300

Name	Description	Chef Type	JSON
chef-load-role-544188300	auto generated role	role	Chef::Role

Details

Attributes

Run List

_default

+ Expand All

- Collapse All

Edit

	Version	Position
+ chef-load-role-544188300
aix::nim_master_setup	2.4.6	...
aix::nim_master_setup_standalone	2.4.6	...
audit	9.5.1	...

With the completion of this work, Chef Infra users no longer need to use Chef Manage [\(which is set to be deprecated at the end of 2022\)](#) and can manage all their infrastructure using Chef Automate, which provides better visibility and manageability. In addition, Chef Automate also provides Chef Infra users with:

- **Role-Based Access Controls:** Provide specific rights to different members of your team and create projects that group users and resources for fine-grained control.
- **APIs and Integration Abilities:** Integrate notifications and data feeds with tools you already use, like ServiceNow, Slack, Elasticsearch, and Web hooks.
- **Built-in Infrastructure tasks** for managing node run_lists, tags and attributes

To learn more about this feature view the [Chef Automate Infra Server documentation](#).

Windows Patching

Chef Infra Client 18 includes expanded security and patching capabilities with the addition of two new resources:

- **WSUS management:** windows_update_settings resource for managing Windows Update settings including WSUS settings
- **Windows Defender Management:** windows_defender and windows_defender_exclusion resources for managing Windows Defender scanning and exclusions

CASE STUDY:



[Tesco Streamlines Patch Management with Self-Checkout for App Teams](#)

Support and Security Updates

The most immediate reason to keep your clients updated is to maintain support and ensure the most up-to-date security patches. While new feature updates will be limited to the latest release, security patches and bug fixes are provided for new and legacy fixes over their entire support lifecycle.

We always recommend Chef Infra users run the latest version of the Chef Infra Client. Updates, published as minor or patch releases, are designed to be non-breaking, backwards compatible, and most importantly, used without updating associated cookbooks. Each Chef Infra Client release typically features performance improvements, underlying component upgrades, additional platform support for new operating systems like AlmaLinux, and timely updates and patches that help you better respond to any nascent vulnerabilities or CVEs in any Chef Infra's dependencies.

Chef provides support, bug fixes, and security patches for the latest two major releases of Chef Infra Client. That means Chef Infra Client 17 will continue to be supported, but Chef Infra Client 16 is officially End of Life as of this release (April 28, 2022).

More information and a full list of software Chef supports can be found in [our documentation](#).

Chef Infra Client Release History

Below is a summary of key features released from Chef Infra Client 18 to Chef Infra Client 12. Note with the release of Chef Infra Client 18, Chef Infra Client 12-16 are now end of life (EOL).

Chef Infra Client 18

- Slow Resource Reporting with `--slow-report` flag
- Ability to allow or deny attributes that are reported to Automate
- Secrets Management Beta
- Run Lists can now be specified when using Policyfiles
- Chef Solo recipe fetching for S3 using `--recipe-url s3://BUCKET/file.tar.gz`
- Improved VMware detection
- Updates to Windows, macOS resources
- Improved Compliance Phase integration between Chef Infra and Chef InSpec

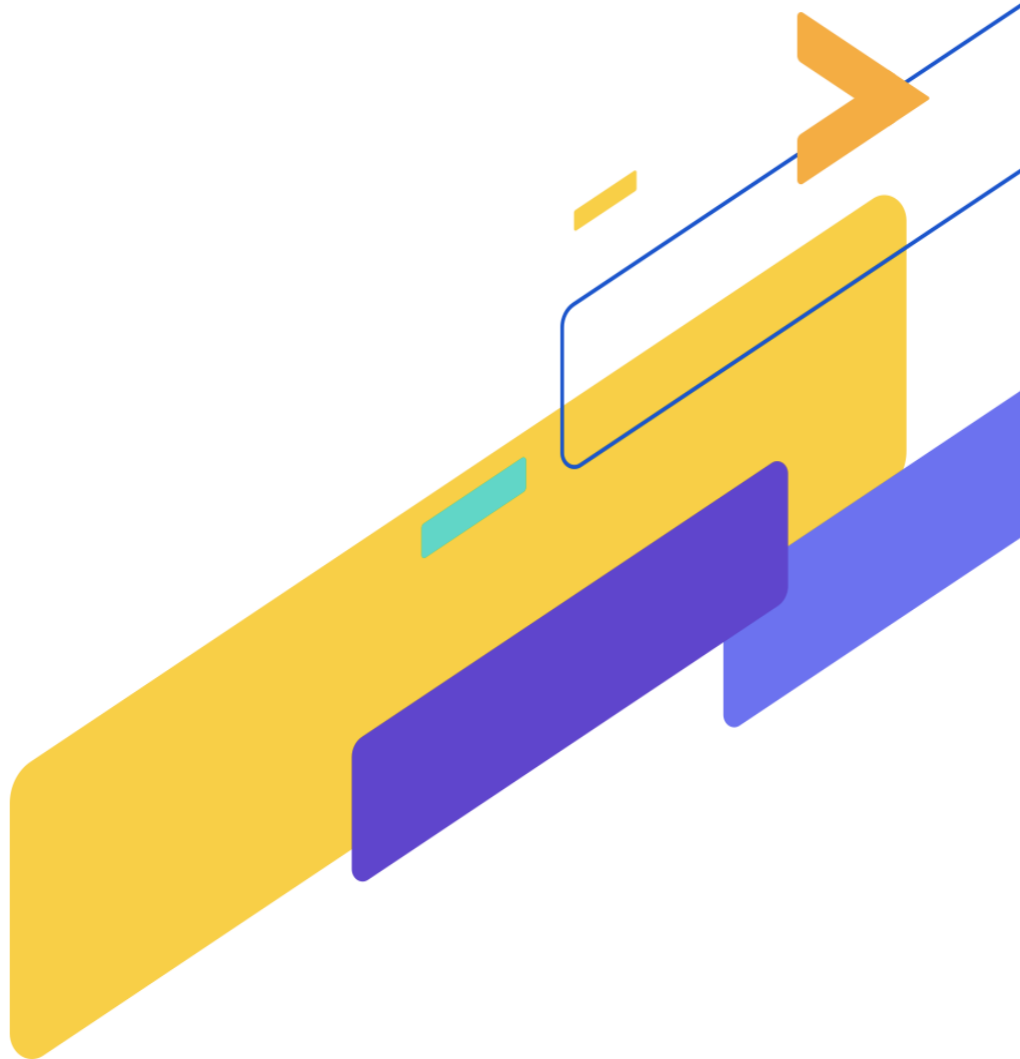
Chef Infra Client 17 (to 17.5)

The release of Chef Infra Client 17 includes a number of upgrades and improvements, including a renewed focus on Test Driven Development (TDD), enhanced system coverage for different architectures and cloud instances, and more helpers that make using and upgrading Chef easier, whether you're a seasoned pro or just getting started. Release Highlights:

- **General improvements**
 - Chef Infra Compliance Phase
 - Ship InSpec profiles, waivers, and inputs directly in cookbooks
 - New compliance CLI reporter
 - Improved Docker container detection and support for detecting containers in Podman
 - New Ohai Chef Habitat plugin at `node['habitat']`.
 - Smaller, more efficient Chef Infra Client codebase without knife and dependencies bundled with Chef Infra Client. *Note: Knife is still included with every Chef Workstation release*
 - Chef packages on macOS/Linux/Unix now create the `/etc/chef` directory along with sample client configuration files to make getting started easier
 - Reporting on the slowest resources in your Chef Infra Client execution with `--slow-report` command line flag and Test Kitchen integration
- **Chef Workstation improvements**
 - Support for macOS Big Sur, macOS Monterey, Windows 8, AlmaLinux and Amazon Linux 2
 - Improvements to Policyfiles and the Chef command line
 - New Cookstyle cops added to modernize cookbook code automatically.
 - Enhancements to the `chef generate` command to generate better getting started `chef-repo` and cookbook files
- **Chef Infra Language**
 - Upgraded to Ruby 3.0 for improved performance and capabilities
 - Custom resource default values are now better processes to avoid potentially confusing errors
 - Recipe attributes can be lazily loaded if necessary
 - `reboot_pending?` Helper now works on all Debian platform-family distros, not just Ubuntu
 - Beta release of new secrets helper for fetching secrets from AWS Secrets Manager, Azure Key Vault, HashiCorp Vault, and Akeyless Vault
 - New `render_json`, `render_toml`, and `render_yaml` helpers for writing out JSON, TOML, and YAML config files

- **New resources**
 - New resources for defining InSpec compliance waivers and inputs directly in Infra recipes.
 - Six new resources for installing Chef Habitat and deploying/managing Habitat applications using Chef Infra
 - `windows_update_settings` resource for managing Windows Update settings including WSUS settings
 - `windows_defender` and `windows_defender_exclusion` resources for managing Windows Defender scanning and exclusions
 - Expanded support for security policies in `windows_security_policy`
 - Improved Windows printer setup capabilities with updates to `windows_printer` and `windows_printer_port`
 - Supports changing hostnames on AD joined Windows nodes in the `hostname` resource
 - Support for setting firewall rules for multiple IP addresses in `windows_firewall_rule`
 - Improved performance in `systemd_unit` resource
 - Performance in the `file` resource improved when using verifiers
 - `windows_certificate` has improved support for importing certs and the ability to export certificates to disk
- **Expanded platform support**
 - macOS 12 and macOS on M1 processors now supported
 - Detection of Sangoma Linux, AlmaLinux, Virtuozzo, and Alibaba Cloud Linux
 - Large improvement to Linux CPU detection
 - Improved filesystem detection format on Windows hosts to match Linux/macOS/*nix hosts
 - FIPS support on Ubuntu
 - FIPS support on RHEL/CentOS PPC architectures
- **Cloud coverage enhancements**
 - Microsoft Azure: Cloud detection has been improved and Increased data gathering and exposure (e.g., server demographics, geos, security groups, whether or not monitoring is enabled)
 - Amazon Web Services (AWS): Support for AWS metadata services (IMDSv2) and improved AWS metadata gathering
 - Alibaba Cloud support with `node['alibaba']` showing metadata with helper `node['cloud']`

- **Secrets management integration helper (Beta)**
 - AWS Secrets Manager
 - Azure Key Vault
 - HashiCorp Vault
 - Akeyless Vault



Chef Infra Client 16

Chef Infra Client 16 debuted as a smaller, faster version of the chef-client that's chock full of new resources, features, and supported platforms. It had an unprecedented focus on streamlining the user experience for newcomers and seasoned veterans alike. Release Highlights:

- Added YAML recipe support
- File unmask can be set on all resources
- Windows improvements
 - Reduced disk usage by up to 30% with dramatically improved performance on Windows systems
 - PowerShell Core Support
 - Enhanced 32-bit Windows Support
- Windows resources can specify SIDs directly instead of user/group names
- 14 new resources
 - alternatives, plist, user_ulimit, windows_security_policy, windows_user_privilege, chef_client_cron, chef_client_systemd_timer, chef_client_scheduled_task, chef_client_config, chef_client_launchd, chef_client_trusted_certificate, windows_firewall_profile, windows_audit_policy, homebrew_update
- Significant Improvements to 26 existing built-in resources and new helper functions, which can be used in any resource or recipe
 - sanitized_path, which, and more
- Built-in helpers for using Chef Vault secrets
- Ohai Plugin improvements
 - Improved Azure detection
 - AWS Instance Metadata Service Version 2 (IMDSv2) support
 - EC2 IAM role gathering
 - Expanded Linux network configuration gathering
 - New plugins for IPC, SELinux, and Interrupts
 - DMI plugin support for Windows
 - Improved gathering of ZFS pool information
- Custom Resource improvements
 - Unified Mode (single phase execution)
 - Improved property require behavior
 - Resource partials for code reuse between resources
 - New after_resource state
 - Improvements and default behavior for identity and desired_state properties
- The compile_time property is now available for all resources, including custom resources
- Upgraded to Ruby 2.7

Chef Infra Client 15

Chef Infra Client 15 will soon be End of Life (April 2021). It included an update to our licensing policies, in which we made all of Chef's software open source under an Apache2 license, and their supported distributions (binaries) subject to an enterprise license for commercial use. More detail can be found in [this blog post](#). Additionally, this release featured a significant number of new helper functions to help with cookbook creation and expanded support for Chef Infra Client on the ARM architecture. Release Highlights:

- Expanded Platform Support
 - x86_64: Ubuntu 20.04, Debian 10, macOS 10.15 (Catalina), Amazon Linux 2
 - aarch64: Ubuntu 18.04/20.04, RHEL 7/8, Amazon Linux 2, SLES 15
- Support for the DNF packaging system in RHEL 8
- New Resources
 - snap_package, archive_file, windows_uac, windows_dfs_folder, windows_dfs_server, windows_dns_record, windows_dns_zone, chocolatey_feature, chef_sleep, notify_group
- Ohai system information detection improvements
 - Improved detection of Virtualized guests in Ohai
 - Improved detection of Windows running on OpenStack
- New Helpers to simplify writing cookbooks and resources
- Multiple platform detection helpers for cloud, virtualization, and OS version
- Replaced Existing Cookbook Dependencies
 - Resources previously provided by the windows_dfs, windows_dns, and libarchive cookbooks are now built-in
- Support for Ed25519 SSH keys
 - Unified Bootstrapping of *nix/Windows
- Target Mode for node management over SSH without a client installation
- Upgraded to Ruby 2.6

Chef Infra Client 14

Chef Infra Client 14 saw a vast improvement in performance and reduction of the install size. We also added a number of new resources that were previously provided by cookbooks on the [Chef Supermarket](#). With these changes, Chef Infra practitioners not only saw the client itself become easier to manage but could greatly reduce the number of cookbooks they needed. Release Highlights:

- Expanded platform support
 - AIX 7 .2
 - Debian 10
 - FreeBSD 12
 - macOS 10 .15
 - RHEL 8
 - Ubuntu 20.04
 - Windows 10 & 2019
- New Resources
 - windows_workgroup, windows_shortcut, windows_printer_port, windows_printer, windows_font, windows_feature, windows_auto_run, windows_ad_join, sysctl, swap_file, sudo, rhsm_subscription, rhsm_repo, rhsm_register, rhsm_errata_level, rhsm_errata, openssl_rsa_public_key, openssl_rsa_private_key, openssl_dhparam, ohai_hint, macos_userdefaults, hostname, homebrew_tap, homebrew_cask, dmg_package, chef_handler, ssh_known_hosts_entry, kernel_module, powershell_package_source, chocolatey_source, chocolatey_config, openssl_ec_public_key, openssl_ec_private_key, openssl_x509_crl, openssl_x509_request, openssl_x509_certificate, cron_access, cron_d, windows_workgroup, locale, timezone, windows_firewall_rule, windows_share, windows_certificate, and build_essential
- Improved Resources
 - windows_service can now create Windows services
- Removable Cookbook Dependencies
 - windows, build_essential, mac_os_x, openssl, sudo, sysctl, rhsm, homebrew, windows_firewall, swap, hostname-chef, locale, timezone_iii
- Improved FIPS detection
- Install size reduced by 50% on Linux/macOS, 12% on Windows
- Upgraded to Ruby 2.5

Chef Infra Client 13

With Chef Infra Client 13, we established our current yearly major release cadence. Full details can be found in the [Chef Infra Release and Support Schedule](#). As part of this change, any planned deprecations, syntax revisions, or other breaking changes must first be implemented as a non-breaking warning that indicates removal in the next major release. Similarly, while patches, bug fixes, and CVE remediations would continue to be implemented throughout each release, changes that might impact behavior or performance, like Ruby upgrades to the next minor release, would be scheduled for the next major release of Chef Infra Client.

Release Highlights:

- Expanded platform support
 - macOS 10.14
 - SUSE Linux Enterprise Server (SLES) 15
- New Resources
 - `apt_preference`, `windows_path`, `windows_task`, `zypper_repository`
- Ohai Improvements
 - Expanded detection of nodes running on Azure, EC2, OpenStack, and SoftLayer
- Cloud metadata gathering for Azure and Rackspace Replaces Existing Cookbook Dependencies
- Resources provided by the `apt` and `zypper` cookbooks are now built-in
- Encrypted Data Bags use more secure `aes-256-gcm` encryption method by default
- Chef InSpec and Chef Vault included out of the box

Chef Infra Client 12

Chef Infra Client 12 was unique in a number of ways. It was the final release before we formalized the yearly cadence of major releases and was one of the longest-running stable releases of Chef Infra Client. As such, a large number of improvements were added during its lifecycle.

Release Highlights:

- Expanded platform support
 - AIX support added
 - MacOS support added
 - SUSE Linux Enterprise Server (SLES) support added
 - Ubuntu 16.04 / 18.04
 - RHEL 7
 - Debian 8
 - Windows 2016
 - FreeBSD 10 / 11
- Expanded macOS Support
 - New Resources: `homebrew_package`, `osx_profile`
- Other New Resources
 - `bff_package`, `openbsd_package`, `paludis_package`, `apt_update`, `launchd`, `yum_repository`, `ksh`, `systemd_unit`
- Removable Cookbook Dependencies
 - Resources provided by the `yum` and `systemd` cookbooks are now built-in
- Notification Timers
 - Determine when a `notifies` or `subscribes` parameter is executed
 - Supports `:delayed` (default), `:before`, `:immediately`
- Security Updates
 - Client/Server connections over HTTPS by default
 - FIPS Mode added
- [Custom Resources](#) Introduced
- [Policyfiles](#) Introduced
- Chef Automate data collection Introduced

Upgrade Preparation Useful Info

There's a lot of ground to cover when we talk about upgrades, and there are many important concepts that don't necessarily fit into some of our more specific topics. We'll start with an overview of some key concepts that should prove invaluable going forward.

Chef Infra Server Versions

It's important to note that the Chef Infra Client follows a yearly major release schedule, whereas the [Chef Infra Server](#) does not. This structure allows us to continuously improve Chef Infra and provide a predictable timeline for any planned feature updates or breaking changes. It also means **there is no parity between Chef client and server version numbers**.

As of this writing, the latest stable Chef Infra Server release is 14.2.2, and Chef provides support for 13.x and 14.x releases. Either version can be used regardless of the Chef Infra Client version you have installed.

Chef Workstation and ChefDK

[Chef Workstation](#) was introduced as a drop-in replacement for the ChefDK. Chef Workstation contains the same tools along with additional enhancements like ad-hoc task support with chef-run and a desktop application with auto-update facilities. Major development of ChefDK ended in 2019 and **on Dec. 31, 2020, ChefDK reached end-of-life status**. If you're still using ChefDK, check out our blog post [Goodbye ChefDK, Hello Chef Workstation](#).

Also of note, regardless of the version of Chef Infra you're using, Chef recommends running the latest version of Chef Workstation. Tools like Cookstyle and Test Kitchen provide built-in facilities for targeting specific Chef Infra Client versions, so upgrading Workstation will not require you to upgrade Chef Infra Client as well.

Upgrading Cookbooks & Clients

Upgrading Chef Infra Client is a twofold process:

1. **Updating Chef cookbooks to ensure compatibility**
2. **Upgrading Chef Infra Client itself on managed systems**

How much planning and work is required for each task will vary from organization to organization. Some may find that their cookbooks need very little work and can focus their efforts on orchestrating client upgrades. Some will have large numbers and combinations of cookbooks in place across environments and platforms and look to subdivide the upgrade process to keep the process manageable. In the next section, we'll address these tasks individually, and provide tips & tricks for each.

Upgrading Cookbook Guidance and FAQ

This guidance is specific for upgrading your Chef cookbooks, which is often done in conjunction with a major Chef Infra Client upgrade. Specifically, we'll cover how to evaluate whether your cookbooks are compatible with the latest versions, and how to use the tools provided in Chef Workstation to quickly iterate on and test those cookbooks against a target release.

Chef Upgrade Lab

Chef has taken what its learned from helping our clients upgrade cookbooks and clients at scale and poured them into [Chef Upgrade Lab](#). Upgrade Lab provides a way to evaluate every cookbook and node on your Chef Infra Server to ensure a straightforward path to migrating everything to the latest and greatest kit.

How do I know which issues affected me?

In the past, you needed to download multiple versions of ChefDK that mapped to the Chef Infra Client version you were going from, to the version you were going to. Then you'd run `foodcritic` against each cookbook and start traipsing through the output.

With [Chef Cookstyle](#), that's not a thing anymore! You can simply set a target version in your cookbook's `.rubocop.yml` (in this example we'll be going from 12 to 13) and get rolling! It's important to note when you specify a version constraint like we've done, that only the warnings specific to this version will be displayed. When you change the target version or the release specification, you'll have more to fix. Please check out the [Chef Cookstyle documentation](#) for a complete review!

If you'd like to try an upgrade on a test repository to see some errors, first clone this repo (github.com/ChefRycar/bad_dates) and make sure you have the latest version of Chef Workstation installed (downloads.chef.io/chef-workstation/).

To learn more about upgrading with Chef Cookstyle [read the blog and watch the recorded webinar: "Chef Infra Best Practices: #1 Using Cookstyle for Infra Client Upgrades"](#).

Which offenses do I fix first?

The ones that Cookstyle can autocorrect of course! What's that? Cookstyle has an autocorrect feature in it that fixes many issues automatically. We'll take a look at that below, but first let's see which deprecations exist in our cookbook and perform a frequency analysis.

To check offenses that exist when upgrading from Chef Infra Client 12 to 13.latest:

```
$ cd ~/path/to/bad_dates > cookstyle .
```

You'll now see output that looks like this:

```
/tmp/bad_dates# cookstyle .
```

```
Inspecting 9 files
```

Offenses:

```
metadata.rb:8:1: R: ChefSharing/InsecureCookbookURL: Insecure
http Github or Gitlab URLsfor metadata source_url/issues_url
fields
issues_url
'http://github.com/ChefRycar/bad_dat
es/issues'
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ *snip*
recipes/windows.rb:13:10: W:
ChefDeprecations/ChocolateyPackageUninstallAction:
Use the:remove action in the chocolatey_package
resource instead of :uninstall which was removedin
Chef Infra Client 14+ action :uninstall
^^^^^^^^^^^^
9 files inspected,
18 offenses
detected
```

Now let's do a quick frequency analysis:

```
/tmp/bad_dates# cookstyle . --format offenses 9/9 files
|=====
100=====
=====>| Time: 00:00:00

3
ChefDeprecations/ChefRewind
3
ChefDeprecations/DeprecatedY
umRepositoryProperties
2 ChefSharing/InsecureCookbookURL
1 ChefCorrectness/NodeNormal
1 ChefDeprecations/ChefWindowsPlatformHelper
1 ChefDeprecations/ChocolateyPackageUninstallAction
1 ChefDeprecations/CookbookDependsOnCompatResource
1
ChefDeprecations/CookbookD
ependsOnPartialSearch 1
ChefDeprecations/CookbookD
ependsOnPoise
1 ChefDeprecations/EpicFail
1
ChefDeprecations/Node
Set 1
ChefDeprecations/Wind
owsTaskChangeAction 1
Style/StringLiterals
--
18 Total
```

Now, that's useful information! At this point if we really want to dive in, we can compare the output from the analysis with all of the rules that currently exist in Cookstyle (github.com/chef/cookstyle/blob/master/docs/cops.md).

For the sake of brevity, we won't do that here. Instead, we'll look at the autocorrect logic.

To use Cookstyle to autocorrect your cookbook, add the -a flag:

```
/tmp/bad_dates# cookstyle -a .
```

```
Inspecting 9 files
```

Offenses:

```
metadata.rb:8:1: R: [Corrected]
ChefSharing/InsecureCookbookURL: Insecure http
Github orGitlab URLs for metadata
source_url/issues_url fields issues_url
'http://github.com/ChefRycar/bad_dates/issues'
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
^^^^^^

metadata.rb:9:1: R: [Corrected]
ChefSharing/InsecureCookbookURL: Insecure http
Github orGitlab URLs for metadata
source_url/issues_url fields source_url
'http://github.com/ChefRycar/bad_dates'
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

*snip*

recipes/windows.rb:9:11: W: [Corrected]
ChefDeprecations/WindowsTaskChangeAction: The:change
action in the windows_task resource was removed when
windows_task was added toChef Infra Client 13+. The
default action of :create should can now be used to
create an update tasks. action [:change, :create]
^^^^^^ recipes/windows.rb:13:10: W:
[Corrected]ChefDeprecations/ChocolateyPackageUninstallAc
tion: Use the :remove action in the chocolatey_package
resource instead of :uninstall which was removed in Chef
Infra Client14+ action :uninstall
^^^^^^^^^^

9 files inspected, 18 offenses detected, 15 offenses corrected.
```

From 18 offenses down to 3 in only a few seconds! This will clearly scale.

Will correcting these offenses break my current chef-client runs and/or pipeline build runners?

This is by far the most common and important question our customers ask us with respect to the upgrade process. When working through the cookbook version and client upgrades, we recommend you perform them one major version at a time. In this example, we're moving from 12.x compatibility to 13.latest compatibility. By doing this, you're introducing code changes required in 13 that are supported in 12, thus giving you the confidence and control to upgrade cookbook by cookbook instead of all at once. Once your codebase has full compatibility with version 13, you can seamlessly upgrade the Chef Infra Client version running in your estate.

Once you're completely stable on the version of Infra Client that you just moved to, you repeat the process from 13 => 14, 14 => 15, 15 => 16, 16 => 17, and finally 17 => 18. A quick and easy way to do this is by modifying the `TargetChefVersion` attribute in the `.rubocop.yml` in your cookbook to the version of the client you're intending to upgrade to.

Can I upgrade from Chef Infra Client 12 to 18?

Yes, you can. But because you'll no longer have backward compatibility with your codebase, you'll need to introduce the new code and new client version in an extremely controlled fashion. This can be done, but it is not advised unless you've had your plan thoughtfully planned out and vetted by one of our Progress Customer Architects.

How should I release these changes in a controlled fashion?

Let's continue with the sample repo provided. By now, you should still have two offenses present in the `bad_dates` example cookbook.

To take a look at how to correct what's left, run the following command without the offenses formatter and make the necessary changes:

```
/tmp/bad_dates# cookstyle .  
Inspecting 9 files .WR.....
```

Offenses:

```
metadata.rb:12:1: W:
ChefDeprecations/CookbookDependsOnPartialSearch: Don't
depend on the deprecated partial_search cookbook made
obsolete by Chef 13 depends 'partial_search'
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ recipes/default.rb:9:1: R:
ChefCorrectness/NodeNormal: Do not use node.normal.
Replacewith
default/override/force_default/force_override
attribute levels. 12 node.normal['antipattern'] = true
^^^^^^^^^^^^^^

9 files inspected, 2 offenses detected
```

You'll now see we have a couple of offenses that weren't autocorrected by Cookstyle. However, you're always given a path forward or an explanation for what's presented. In this case, we see `partial_search` is remediated by moving to Chef Infra Client 13, which is exactly what we're doing! We also see a warning against using `node.normal` and should defer to our attribute precedence documentation to choose a way forward.

Incredibly important note here:

Once you've used Cookstyle's autocorrect feature, you should absolutely, without exception, run your cookbook through your full suite of local and integration testing using tools like Test Kitchen and its integration in your CI/CD pipeline.

Once the offenses are corrected and after a series of successful integration tests, you have a cookbook that is completely compatible with Chef Infra Client 13. Above all, it's backwards compatible with your existing Chef Infra Client 12 nodes. You can now begin releasing this cookbook into your pre-production environments with the goal of promoting it to your production environment. After your cookbook code is sorted, you may now upgrade the Chef Infra Client itself to the target version. Rinse and repeat until you're at Chef's latest kit!

How can I handle this in the future?

Like all things in life, you get better with practice. The goal should be to never let yourself get more than 1 major version behind what is currently supported by Chef. This starts with shoring up how you perform local development.

Once you're on Chef Client 16/17/18 and the latest Chef Workstation, you can use Test Kitchen attributes (https://docs.chef.io/workstation/config_yaml_kitchen/) to be declarative about the versions you want to write code for. Adding the following to your kitchen.yml will ensure your cookbook is compiled properly on the target version, and deprecations will present as errors, forcing developers to keep their code compatible.

In your kitchen.yml:

```
# Set target version (latest by default)
https://docs.chef.io/workstation/config\_yaml\_kitchen/#new-provisioner-
settingsproduct version: latest

# Set deprecations as errors
(https://docs.chef.io/workstation/config\_yaml\_kitchen/#provisioner-
settings)

provisioner:
  deprecations_as_
  errors: true
```

You can also have Cookstyle autocorrection run automatically in local development or add it into your CI/CD pipeline's linting/correction phase.

Using these simple methods will enable you to correct deprecations and offenses as part of the natural lifecycle of your codebase, rather than allowing it to become a monumental effort.

Upgrading Chef Infra Client Guidance

Once you've updated your cookbooks, and ensured compatibility with Test Kitchen, it's time to start upgrading clients in your environments. Before starting any upgrade process, be sure to check out the upgrade considerations in [our documentation](#).

After the chef-infra-client upgrade, you will see the following benefits:

- Vulnerability and security patches (years of patched vulnerabilities and exposures)
- Utilization of a supported release
- Access new features and resources
- Bug fixes and general speed improvements

The Manual Path

Upgrading something that potentially manages your entire infrastructure can appear burdensome. While there are more automated ways of dealing with this task, it's important to understand how to approach it manually. The manual way of upgrading the chef-client involves running a CLI command to upgrade on a node-by-node basis.

The following line would do the trick on *nix and macOS:

```
$ curl -L https://chef.io/chef/install.sh | sudo bash
```

And here's the Windows one-liner:

```
$ . { iwr -useb https://omnitruck.chef.io/install.ps1 } | iex;  
install
```

While having to SSH or WinRM into machines and [upgrading via command-line](#) is not extremely challenging, you can run into several problems:

- Specifying different Chef Infra Client versions
- Dealing with different native OS processes
- Time costs of upgrading at scale

For these very reasons, we've created the chef_client_updater cookbook, which does most of the heavy lifting for you.

The Accelerator chef_client_updater

The chef_client_updater allows you to upgrade your Chef Infra Client through a cookbook. This recommended approach leads to an easy-to-digest process that gives you the following benefits:

- Upgrading to any desired Chef version
- In general, binaries expect to be upgraded one minor release at a time. This is generally done so functionality remains throughout the upgrade process.

- You've already taken care of functionality through the Cookstyle/refactoring exercise, which means you can jump to whatever major version you desire.
- Through the `chef_client_updater` you can simply pin the version of Chef Infra Client you want to migrate to and press play.
- Upgrading at Scale

Your current Chef Infra consumption may translate to the challenge of having to upgrade hundreds or thousands of nodes.

Thankfully Chef lives for Automation!

If you're already utilizing Chef Infra at that level of scale, then your nodes are probably already communicating with some version of the Chef Infra Server. By adding the `chef_client_updater` cookbook to your run-list and pushing it to the Chef Infra Server, your nodes will auto-upgrade the chef-client the next time they self-check for desired states.

This upgrade will only happen if the Chef Infra Client installed version doesn't match your desired version. This means that the setup process only needs to happen once, and Chef Infra handles the rest.

The entire process is fairly fast and after you execute it once, you can apply it to whatever scale you want. Simply add the `chef_client_updater` as one of your dependencies and use the resource like so:

```
chef_client_updater 'Install latest Chef Infra Client 18.x' do  
  version '18' end
```

[Here's a video of performing a quick upgrade on a Windows machine:](#)



Additional Resources

- Chef DevRel Team Live [Twitch Stream](#)
- Chef Infra Best Practices [Webinar Series](#)
- Chef Infra Client [Upgrade Documentation](#)
- The [chef client updater](#) Cookbook
- The [Learn Chef](#) experience
- Chef [Product Announcements](#)
-

As ever, if you'd like to upgrade but could use assistance, be sure to [contact us](#) to see how we can help jumpstart your upgrade process!

About Progress

Dedicated to propelling business forward in a technology-driven world, Progress (NASDAQ: PRGS) helps businesses drive faster cycles of innovation, fuel momentum and accelerate their path to success. As the trusted provider of the best products to develop, deploy and manage high- impact applications, Progress enables customers to develop the applications and experiences they need, deploy where and how they want and manage it all safely and securely. Hundreds of thousands of enterprises, including 1,700 software companies and 3.5 million developers, depend on Progress to achieve their goals—with confidence. Learn more at www.progress.com.

-  facebook.com/getchefdotcom
-  twitter.com/chef
-  youtube.com/getchef
-  linkedin.com/company/chef-software
-  learn.chef.io
-  github.com/chef
-  twitch.tv/chefsoftware

© 2022 Progress Software Corporation and/or its subsidiaries or affiliates.
All rights reserved. RITM0180258