chrome enterprise

Updates technical document

chrome enterprise

Contents

- Introduction
- Common update management strategies
  - Strategy 1: Auto-update
  - Strategy 2: Version pinning by milestone
    - Leveraging the extended stable channel
  - Strategy 3: Version pinning by full version
  - Strategy 4: Full manual updates
- Best practices
  - Stay up to date with the Enterprise release notes
  - Ensuring that your users restart Chrome to apply updates
  - Leveraging the Beta channel
  - Ensuring that your version reports on Chrome are accurate
- Understanding Chrome update mechanisms
  - Google Update
  - Chrome Variations Framework
  - Component Updates
- Other considerations
  - Working with limited bandwidth
  - Set up maintenance windows
  - Stagger your updates
  - Cache updates
  - Dealing with a bug or incompatibility
  - Rollback
  - Understanding why Chrome isn't updating to a new version
  - Troubleshooting
    - Gather logs
    - Common Update issues
    - Fixing updates on Macs
    - URL allowlist
    - Does Chrome get updates when it’s not running?
- Conclusion
  - Further information

Introduction

Chrome's default behavior is the most secure and the easiest to manage, but Chrome still provides a range of controls to help IT admins balance security and stability in their organization. If you're an IT administrator and have specific requirements not met by Chrome's default settings, this technical document is for you.

We will outline four common update management strategies, along with the tradeoffs each of them have. We will then outline some best practices applicable to all update strategies. Finally, we'll explain some of the more nuanced parts of Chrome's update mechanisms in detail.

Note that extensions are updated via a separate process, which is explained in our Extensions Management technical document.

chrome enterprise

Common update management strategies

How should you manage Chrome updates? To answer that, we first have to decide on our goals. Let's look at three goals in particular:

1. **Security—** This is the primary consideration of most enterprises, and it's ours too. A good update strategy keeps your users secure.
2. **IT effort—** We know you're busy. A good update strategy should be easy to implement so you can concentrate on the important things.
3. **Stability—** Your users need to stay productive. A good strategy minimizes the risk of bugs and bad interactions between the different pieces of your environment. It helps you stay ahead of breaking changes, and react swiftly if anything goes wrong.

Let's look at some common update strategies and see how they do against these goals. This table summarizes them:

Update strategy	Security	IT effort	Stability
Auto-update	Best	Best	Standard
Version pinning by milestone	Medium	Medium	Better
Version pinning by full version	Worst	Medium	Better
Full manual updates	Worst	Worst	Better

In this section, we'll describe each of these strategies in detail.

Strategy 1: Auto-update

This is the recommended best practice, and Chrome's default behavior. With auto-update, new versions are automatically downloaded by Google Update and applied when users restart their browsers.

Pros

- **The most secure strategy—** Updates frequently contain security fixes and should be applied quickly (within days or weeks). Although many enterprises concentrate on applying fixes to known in-the-wild exploits, other security fixes are just as important. Once a fix is released, bad actors may reverse engineer it, turning into an in-the-wild exploit after the release.
- **The least effort—** No need to manually deploy each release/security patch or centrally manage them; the browser will update itself.
- **Always on a supported version of Chrome—** Enterprises that have support will always have the latest fixes.

Cons

- **Requires intervention for breaking changes—** There is some risk that a bug or planned breaking change will affect your users' productivity. You may have to occasionally intervene to reinstate the old behavior or rollback Chrome's version.

Configuring auto-update

Since this is the default behavior, no policies need to be set. However, you may need to allow Google Update to communicate through your proxy or firewall. A list of URLs that are used for Chrome updates can be found here: Manage Chrome updates

If you want to configure auto-update explicitly (optional):

Via GPO: Set **Google > Google Update > Applications > Google Chrome > Update policy override** to **Always allow updates**.

Via the admin console: (Windows only): Set **User & browser settings page > Chrome updates section > Chrome browser updates** to **Allow updates**.

Strategy 2: Version pinning by milestone

Chrome does extensive testing before any changes are rolled out to the Stable channel, but enterprises frequently have unique environments and custom software that interacts with Chrome in unpredictable ways. Sometimes, a change in Chrome exposes an underlying bug somewhere else in your environment. Version pinning by milestone may be appropriate for particularly complex environments with little tolerance for disruptions.

Chrome has a new milestone release every 4 weeks on the Stable channel, and every 8 weeks on the Extended Stable channel. Milestone releases increase the first number in Chrome's version string—e.g. **101**.0.4951.67 to **102**.0.5005.61. Any code that adds functionality, removes or materially changes functionality, or risks interacting poorly with other parts of your environment will be introduced in a milestone release.

Chrome has minor releases between milestones. These minor releases generally contain security improvements or fix functional regressions. They have a much lower risk of introducing bugs or breaking changes. Their version string starts with the same number as the previous version—e.g. **102**.0.5005.61 to **102**.0.5005.62.

Version pinning to a milestone allows Chrome to update to a new minor release automatically, but milestone releases are only applied after you've tested and approved them.

Pros

- **Improved stability, even for complex environments—** Code changes that risk introducing a bug or breaking change do not go out to your users until you've tested and approved them.
- **Security releases and functional fixes applied automatically—** This strategy is less work and safer than strategies 3 and 4 discussed below. You only need to approve the major releases that happen at regular intervals and contain the most risk.

Cons

- **Some security improvements are delayed—** Some security improvements cannot be introduced without the risk of interacting poorly with other parts of your environment, and so are introduced with a new milestone. Since Chrome will wait for your approval before updating to a new milestone, these improvements will be delayed.
- **Maintaining a test suite—** This strategy expects you to maintain and run tests before approving each milestone, otherwise there's minimal value in approving them.

Configuring version pinning by milestone

You can use the **Target version prefix** policy to specify any major milestone of Chrome. For example, if you want to keep users on version 100 of Chrome, configure this policy to **100.** (including the decimal point). This allows Chrome to take all updates with a version string beginning with "100." only. When milestone 101 is available, Chrome will stop taking updates until you do any necessary testing, then update the policy to **101.** (again, including the decimal point).

Once you've updated the policy, Chrome will update to milestone 101, and will take minor updates within the 101 milestone automatically.

Via GPO: Set **Google > Google Update > Applications > Google Chrome > Target version prefix override** to the specified milestone number and period.

- Example value to pin to milestone 100 = **100.**

Via the admin console (Windows only): Set **User & browser settings page > Chrome updates section > Target version prefix** to the specified milestone number and period.

For Mac: Set the **TargetVersionPrefix** policy to the specified milestone number and period.

Important: If you set **Target version prefix**, remember to regularly update it. If you don't allow the next milestone to roll out, you won't receive any more security updates.

Leveraging the extended stable channel

This strategy works well with the **extended stable** channel. Extended stable updates to a new milestone every 8 weeks instead of every 4 weeks, which reduces the amount of testing you need to do. Extended stable receives security fixes over its entire 8 week lifespan.

If you've installed Chrome using the Stable binary, you can specify which channel Chrome follows on Windows or Mac by configuring **TargetChannel** to **extended**.

In the admin console, you can set the channel Chrome on Windows using **User & browser settings page > Chrome updates section > Chrome Browser updates > Channel dropdown**

Strategy 3: Version pinning by full version

You can also pin to a specific version. With this strategy, Chrome will not take any updates until you've approved them—not even minor updates. This strategy may be appropriate for highly locked down environments with no tolerance for risk and where lots of resources can be devoted to regularly testing each new version of Chrome before it's rolled out.

While new milestones happen predictably every 4 or 8 weeks, minor updates happen much more frequently, and are released without notice. This allows us to quickly issue urgent security updates (for example, 0-day fixes), and fix regressions caught early in a rollout. The rollout of a milestone may involve multiple minor releases. Because managing and testing each individual minor release is arduous and unpredictable, we recommend pinning to milestones instead (see Strategy 2).

Pros

- **No Chrome updates without your approval—** You have the opportunity to test every update before it's rolled out

Cons

- **Maintaining a test suite—** This strategy expects you to maintain and run tests before approving each update, otherwise there's minimal value in approving them.
- **Less secure than milestone pinning—** Because Chrome won't take any security updates without your intervention, security fixes take longer to reach your users. If you don't update **Target version prefix** each time there's a Chrome release, your users will be stuck on an unsupported version.
- **More work than milestone pinning—** Under this strategy, you need to take an action every time a Chrome update is available. This can be unpredictable, and happens much more frequently than milestone releases.

Configuring version pinning by full version

You can use **Target version prefix** to specify a full version string, e.g. "**100.0.3987.158**". In this case, Chrome will update to the chosen version string, bypassing any gradual ramp up Google has set. Because this configuration skips the safeguards Chrome uses to minimize the risk of breakage, this should only be done after you test the new version on your environment.

Configuring version pinning by milestone explains how to set **Target version prefix**.

Strategy 4: Full manual updates

Some organizations run Chrome in locked-down environments where there is no internet access, and the browser is used for internal webapps only. In this environment, Google Update is not an option, and you must update Chrome by manually pushing a new MSI each time.

If Chrome has internet access, bear in mind the risks of a fully manual approach, and minimize the number of users who are updated in this fashion. Without access to automatic updates, browsers can miss critical fixes, leaving them susceptible to vulnerabilities. Although many enterprises concentrate on applying fixes to known in-the-wild exploits, other security fixes are just as important. Once a fix is released, bad actors may reverse engineer it, turning into an in-the-wild exploit after the release.

Consider Strategy 2 instead. It provides many of the same stability benefits, while being less work and more secure.

Note that even with this approach, there is no need to uninstall Chrome before installing a new version—you can simply push the new MSI to all the machines you wish to update.

Pros

- **Works for devices that have no internet connection—** This is the only strategy for devices that have no access to the internet.

Cons

- **Less secure than all other strategies—** This strategy has the biggest delay in rolling out updates and fixes, and has the most risk of users being stuck on an unsupported version of Chrome.
- **More work than all other strategies—** This disables Google Update and requires the IT team to recreate the same functionality from scratch.

Configuring full manual updates

Via GPO: Set **Google > Google Update > Applications > Google Chrome** to **Disable updates**.

In the admin console, you can set the channel Chrome on Windows using the **User & browser settings page > Chrome updates section > Chrome Browser updates > Disable updates.**

Best practices

These best practices apply no matter which of the above strategies you're using.

Stay up to date with the Enterprise release notes

To know what's updated in each Chrome milestone, subscribe to the Enterprise Release Notes.

Ensuring that your users restart Chrome to apply updates

As long as the machine is powered on, has network connectivity, and Google Update has not been disabled by policy, Chrome will be updated silently in the background when a new update is available. However, if your users keep Chrome open, it will stop the update from applying until they restart. Chrome will display a hint in the top right of the window to remind users to restart and update automatically.

You can display more visible reminders by setting **RelaunchNotification** to **Recommended** and configuring **RelaunchNotificationPeriod** to specify the time period for notifications (default is one week, and the minimum is one hour).

To force a relaunch rather than merely recommending it, set **RelaunchNotification** to **Required**, and specify the time period before relaunch using **RelaunchNotificationPeriod**. The minimum time period you can specify is 1 hour (3600000 milliseconds), and the default is one week (168 hours, or 604800000 milliseconds). Note that in the Admin Console, the relaunch notification period is specified in hours rather than milliseconds.

These policies only need to be set once, and they will apply for all future updates. This makes it easy for you to determine what tradeoffs make the most sense for your users, configure Chrome the way you need it, then allow Google Update to do the work for you.

Leveraging the Beta channel

If your organization has tests for your critical workflows, run them regularly on Chrome's Beta channel to ensure there's no surprising changes or interactions between Chrome and other pieces of your software stack.

If you don't have tests for your organization's specific workflows, running a subset of users on Beta is another option. Consider putting **5% of users** on the **Beta Channel**. Beta is our release candidate and carries minimal risk of issues. Google fully supports it. Beta users should be spread across a range of functions, to maximize the chances that any issues or incompatibilities arising in Beta are caught before that version moves to Stable. Windows and Mac users can run Beta and Stable side by side, so users can easily switch to Stable Chrome in the unlikely event that a serious issue prevents them from continuing their work in Beta.

If you've installed Chrome using the Stable binary, you can specify which channel Chrome follows on Windows or Mac by configuring **TargetChannel** to **stable**, **extended**, **beta**, or **dev**.

In the admin console, you can set the channel Chrome on Windows using **User & browser settings page > Chrome updates section > Channel dropdown**

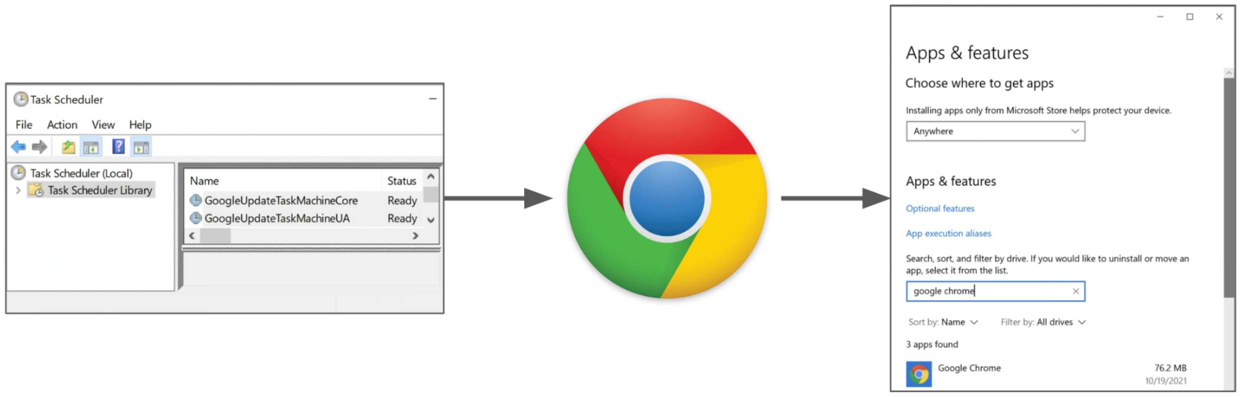
The Beta, Dev, and Canary binaries are locked to their respective channels.

It's also useful to enable MetricsReportingEnabled in order to collect usage statistics for these users, making it easier for Google to detect and fix issues in Beta. Note that even if you enable metrics, users can turn it off if they choose.

Ensuring that your version reports on Chrome are accurate

This best practice is only applicable if you're using a tool to report on which versions of Chrome are installed on Windows devices.

When auto-update is configured and working correctly, Chrome browser will automatically update, even if Chrome is not launched. This happens via the Google update service that runs separately from the Chrome application. However, the version showing in **Add & remove programs** isn't updated until Chrome is launched. Most enterprises (including Chrome Browser Cloud Management) pull reports on what version is present via **Add & remove programs**. So if Chrome hasn't been launched since updating, Chrome will be up to date, but reports might be reporting an old version.



Name	Status
GoogleUpdateTaskMachineCore	Ready
GoogleUpdateTaskMachineUA	Ready

Installing apps only from Microsoft Store helps protect your device.

Anywhere

Search, sort, and filter by drive. If you would like to uninstall or move an app, select it from the list.

google chrome

Google Chrome 76.2 MB 10/19/2021

Example of the workflow of how Chrome updates. Google update updates Chrome, Chrome updates Add & remove programs

A method for getting more accurate reports from those machines where Chrome is not regularly launched is to launch those machines via cmd line in Headless mode. Headless mode is a method for launching Chrome with no UI. So via a simple script, you can launch Chrome via this method, leave it open for a few minutes so the application can update **Add & remove programs** and then close Chrome. This (in combination with the relaunch notification feature for users that don't regularly close Chrome) will greatly increase the accuracy of your reports.

Understanding Chrome update mechanisms

Google Update

Google Update (on Windows) and Google Software Update (on Mac) distribute automatic updates to Chrome and other Google products. Google Update is included in the Chrome installers, so there is no need to install it separately.

Google Update controls and updates the version of the browser; it implements whichever strategy you choose from the sections above.

However, there are a few other mechanisms to be aware of that can change Chrome's behavior, even within a given version. Each of them has their own enterprise controls:

The initial Chrome Browser installation is approximately 56 MB.
- Subsequent updates from one version to the next are approximately 10–15 MB.
- Patch updates are typically 0.5–3 MB.

Updates from a major version to a later non-consecutive major version usually requires a new complete installation.

Chrome Variations Framework

Features and fixes can be gradually enabled (or if necessary, rapidly disabled) via the Chrome Variations framework. The benefit of this approach is that it allows Google to:

- Give a small group of users previews of new features and gather feedback.
- Safely roll out changes to a controlled percentage of users, to minimize the risk of incompatibilities.
- Provide security and other critical updates to you faster.
- Rollback features if needed, without you having to wait for a new version of Chrome. The user only needs to restart their computer to get a new configuration.

If you test a specific version of Chrome before rolling it out to your users, and want to ensure that Chrome's behavior doesn't change between your tests and your production environment, configure ChromeVariations to **Critical fixes only** for both your test and production environments. This disables gradual rollouts of new features, while still allowing Chrome to make emergency changes if necessary (for example, to protect you from an active exploit).

You can disable Chrome Variations altogether by setting ChromeVariations to **Variations disabled**. This option is not recommended, because it stops Chrome from making any changes in any situations (even, for example, as a response to an active exploit).

Component Updates

Some small components of the browser are managed and updated independently of the main Chrome application. For example: models that help classify pages for Safe Browsing, and configuration of active origin trials.

ComponentUpdatesEnabled can be disabled to prevent these updates. This isn't recommended; just like regular Chrome updates, Component updates help keep Chrome safe and stable (by keeping Safe Browsing models up to date, for example).

Other considerations

Working with limited bandwidth

If some of your users work in an environment with limited bandwidth, having them all update their browsers at once may create a heavy demand on the network that can impact their productivity. You can configure Google Update to update Chrome (and any other software it manages) during scheduled maintenance windows, stagger updates over a period of time, or cache updates locally, to keep users with limited bandwidth productive while keeping their browsers up to date.

Set up maintenance windows

Maintenance windows ensure that Chrome updates only take place outside of designated hours, minimizing disruption to your users in their busiest hours. You can specify hours during which Chrome *will not* auto-update by enabling **Time period in each day to suppress auto-update check** and specifying the **Hour** and **Min** of the time each day when you want updates to be suppressed and the **Duration** of time (in minutes) for which they will remain suppressed. Note that the times you specify will be the local machine time, and must be in 24-hour format.

Admin Console (Windows only):
User & browser settings page > Chrome updates section > Suppress auto-update check

GPO:
Google > Google Update > Preferences > Time period in each day to suppress auto-update check

Mac:
UpdatesSuppressedStartHour
UpdatesSuppressedStartMin
UpdatesSuppressedDurationMin

Stagger your updates

Another way to manage updates in a low bandwidth environment is to stagger them so that your entire fleet doesn't update all at the same time. You can do this by specifying a custom time period between update checks, which delays updates to reduce peak bandwidth use.

Admin Console (Windows only):
User & browser settings page > Chrome updates section > Auto-update check period

GPO:
Google > Google Update > Preferences > Auto-update check period override

Note, however, that while delaying updates can help reduce the peak bandwidth use, it may increase total bandwidth use.

To stagger updates, enable **Auto-update check period override** and specify a number between 1 and 43,200 (inclusive) in **Minutes between update checks**.

Cache updates

Admin Console (Windows only):
User & browser settings page > Chrome updates section > Cacheable URLs

GPO:
Google > Google Update > Preferences > Download URL class override

Mac:
DownloadPreference

Chrome updates can also be cached locally using an intermediate proxy cache - most web-caching proxy servers should work. To tell the Google Update server to send Chrome updates via an URL that is more easily cached by Proxy Servers, set **Download URL class override** to **Cacheable download URLs**.

If your proxy server is still having trouble caching Chrome updates, try configuring the following settings:

- **Maximum file object size** - at least 1 GB
- **Cache directory size** - Ensure there is enough storage space either in memory (faster) or on disk
- **URL settings** - Give preference to **dl.google.com/*** and **www.google.com/dl/***
- **Maximum object size in memory** - E.g. 2,000 KB
- **Cache space on disk** - If you have a large hard drive (more than 30 GB), you can increase the value to cache more objects

Setting up a cache in environments with low bandwidth or slow connection speeds can give you better response times, as well as saving bandwidth for other tasks.

Dealing with a bug or incompatibility

If you come across an issue with a specific version of Chrome, after raising a Support Case or a bug, you'll want to update your entire fleet to be sure that all your users receive the fix.

To be sure that all users have received the update, visit the **Versions report** page in the Admin Console. The Versions report page allows you to see all Chrome Browser and Chrome OS versions across your fleet in one place, and you can filter by last active time.

Rollback

In some rare circumstances, you may find it necessary to rollback to a previous version of Chrome while you wait for a fix. To rollback to a previous version, configure **Target version prefix** to the version you'd like to rollback to - this should be the most recent version that works as expected in your environment. You'll also need to enable **Rollback to Target version** for the rollback to take effect.

To ensure that users' data is preserved, please refer to our Help Center documentation on keeping data during version rollback.

Automatic rollback requires automatic updates through Google Update to be enabled, and the device must be domain-joined and/or the browser enrolled in Chrome Browser Cloud Management. You can only rollback to one of the last three versions of Chrome. For browsers that are updated manually, or that need to be rolled back to an older version, you will need to perform the rollback manually.

Admin Console (Windows only):
- User & browser settings page > Chrome updates section > Target version prefix
- User & browser settings page > Chrome updates section > Rollback to target version

GPO:
- Google > Google Update > Applications > Google Chrome > Target version prefix override
- Google > Google Update > Applications > Google Chrome > Rollback to Target version

Mac:
- TargetVersionPrefix
- RollbackToTargetVersion

Understanding why Chrome isn't updating to a new version

You may notice that a version of Chrome exists, but some or all of your browsers are not updating to it immediately, even with auto-updates enabled. There are several reasons why this may happen in the short term (if the new version is less than about a week old), for example:

1. **The new version is not fully rolled out—** Rollouts happen gradually over several days, starting with a random subset of browsers. Depending on circumstances, a rollout can take more than a week. This gradual rollout is one layer of Chrome's strategy to ensure a stable platform. You can see the current rollout percentages at https://chromiumdash.appspot.com/releases.
2. **We may have detected a problem with the new version—** Thus, updates are paused or canceled. In this case, a fixed version will replace the one you've noticed.
3. **A browser has been updated to a "control version"—** Chrome uses multiple new versions of Chrome during rollouts—a control and an updated version. The control version is identical to the previous version and still uses the old milestone number. By comparing control and update versions, we get much better signals to ensure there are no surprises in the updated version. No action is required; as the rollout progresses, browsers updated to a control version will receive another update to the next version.
4. **You're comparing auto-update to the MSI version—** The Chrome download available at https://chromeenterprise.google/browser/download has a unique version string, and Google Update will not serve it automatically to existing browsers. The MSI version is identical to the latest version being served by Google Update. Just like the control version, this facilitates statistical analysis to minimize the risk of issues affecting you.

If that sounds complicated, it's because it is! There's lots of work going on behind the scenes to keep your organization safe and productive, including statistical analysis controlling for confounding variables.

You don't have to manage any of those situations directly. Remember that the best practice is to keep auto-updates enabled, so you don't have to figure out which exact case applies to any specific version. Google Update will manage these tradeoffs and special cases automatically.

You always have control over Chrome's version in your environment. If you set **Target Version Prefix** to a specific version, your choice will take precedence over Google Update's, including bypassing the rollout. Just remember: this skips one of the safety features built into Chrome's update strategy, so you should only do so for versions you've tested internally.

Troubleshooting

Gather logs

If you run into unexpected issues with Google Update, it can be useful to gather logs to help troubleshoot. Logs are also valuable when raising a support case.

Steps for gathering update logs for Windows machines

To create update logs for Mac machines follow these steps:

1. Open up terminal
2. Run
    a. /Library/Google/GoogleSoftwareUpdate/GoogleSoftwareUpdate.bundle/Contents/Helpers/ksdiagnostics
3. It will prompt for admin and dump a .zip file of the update log on the user's desktop.

Common Update issues

Common issues for Google update not functioning correctly usually fall in the following categories:

- Permissions issues
- Network, proxy, and firewall issues (the URLs needed for Google Update to work have been blocked)
- Disk space (there is not enough space to download the update)
- Conflicting group policies (Google update has been disabled)
- Corrupted Google Chrome installation (possible uninstall and then re-install with a profile cache clearing is needed)

Fixing updates on Macs

There is a feature of Keystone (the Chrome Update Agent for macOS) called ksinstall that can help with resolving some update issues that could be caused by end users disabling the update service.

- As of Keystone 1.3.17.192 (released 2022-Apr-18), ksinstall can be configured to detect and remediate the most common ways users disable Keystone.
- Users requesting support for installation errors that appear related to Keystone should be directed to enable this, ensuring that they are on Keystone 1.3.17.192 or higher.

What it does:

- It resets ownership and permissions for files and folders along Keystone’s installation and actives-reporting paths to expected values
- It renames files sitting where Keystone expects to find a directory
- It resets permissions for Keystone’s LaunchAgent/LaunchDaemon configuration plists

How to run ksinstall:

From anywhere: Run these commands from macOS Terminal:

```
defaults write com.google.Keystone.installer nextInstallIsRemedial YES
sudo defaults write com.google.Keystone.installer nextInstallIsRemedial
YES
```

Then attempt the failed installation again. `ksinstall` will reset this flag after its remediation attempt.

When running ksinstall directly: Add the `--remediate` flag to the command line. One example command line might be:

```
sudo ./ksinstall --remediate --force --install=Keystone.tbz
```

but the command will vary depending on the present working directory, where Keystone.tbz is, the desired installation mode, etc.

- The “from anywhere” instructions set a flag in user defaults (for the current user and the root user) that `ksinstall` checks, so this approach also works when running ksinstall directly.
- Remedial mode can be combined with `--uninstall` or `--nuke`, not only `--install`. (If Keystone’s file permissions and ownership got corrupted in a way that breaks uninstallation, this might be useful.)

URL allowlist

Ensure that Google Update can reach the URLs it needs to update Chrome. The Help Center contains a list of URLs to add to your allowlist.

Does Chrome get updates when it’s not running?

Yes!

- As long as the machine is powered on, has network connectivity, and Google Update has not been disabled by policy, Chrome will be updated silently in the background when a new update is available.
- The next launch of Chrome will be the new version.
    - Note that the Admin Console’s Version Report may still report the old version until Chrome is launched.
    - This can be mitigated by filtering the report on Last Activity to remove stale browsers that have not been running for a long time.
    - Third-party tools may not always report Chrome’s version accurately, so check the Version Report or chrome://version on the target device for the most accurate information.

If Chrome is installed at the machine level (rather than at the user level), this will work even if no user is logged in to the device. In either case, the user logged in to the OS *does not* need to have administrator privileges in order for Chrome to update itself.

Conclusion

These are some of the many ways that Chrome gives you control and visibility of your environment. Use these controls to get the best balance of security and stability for your users. For most scenarios, our recommendation is to:

- Keep automatic updates enable via Google Update
- Keep Chrome Variations switched on
- Test in Beta for a preview of the next release
- Subscribe to the Chrome Enterprise release notes

Further information

- Chrome Browser Cloud Management technical document: Get started managing Chrome from the Google Admin Console
- Extensions Management technical document: Details on extension management, including extension updates
- Chrome Enterprise release notes
- Enterprise downloads: Installers and policy templates for Chrome (including the Beta channel) and Google Update
- Automated testing with headless Chrome for developers