

Managing Extensions in Your Enterprise

Securely manage Chrome extensions at scale

Table of contents

Purpose of this guide

Introduction

Considerations for managing Chrome extensions

- What are extension permissions?

- How do extensions update?

Managing extensions

Overview of the different extension management policies

- Block extensions based on their permissions

 - Manage extensions by their permissions in the Chrome Browser Cloud Management

 - Manage extensions by their permissions in Group Policy

 - Creating an exception process for extensions that require risky permissions

Managing extensions by the extensions setting policy

- Configure the extension policy using the Windows Registry

- Configure using a JSON string in Windows Group Policy Editor

- Prevent extensions from altering webpages

Allow or block extensions in the Google Admin console

- Allow all extensions except those you want to block

- Block all extensions except those you want to allow

- Block or allow one extension

- Force-installing extensions

Let users request Extensions: Extension Workflows

Allow or block extensions in Group Policy

- Allow all extensions except those you want to block

- Block or allow one extension

- Force-install an extension

Validating your Policy

Self hosting your own extensions

- Alternatives to self hosting extensions

 - Extension publishing options

- Pin an extension to a specific version in the admin console

- Requirements for self hosting extensions

- Packaging your extension

- Hosting your extension

- Publishing updates to your extension

- Distributing privately hosted extensions

Manage extensions using Chrome Browser Cloud Management

Additional resources

Purpose of this guide

There are many useful extensions built for Chrome browser. There may be many running on your user's computers. This can make controlling and monitoring extensions difficult for IT admins.

This guide is for IT admins who are looking for the best ways to manage extensions. It provides steps for managing extensions using both [Chrome Browser Cloud Management](#) and Windows Group Policies.

This guide is organized by the ways you can manage extensions. You can:

1. Block extensions based on their permissions
2. Manage which websites extensions have access to
3. Allow or block extensions in the Chrome Browser Cloud Management or by Windows Group Policy
4. Self host your own extensions on-premise

What's covered	Instructions and recommendations for managing extensions for Chrome browser in an your enterprise
Primary audience	Microsoft® Windows® and Chrome Browser Cloud Management administrators (Windows, Mac and Linux supported)
Takeaways	Best practices for managing extensions with Chrome browser

Last updated: Nov 19, 2021

Published location: <https://support.google.com/chrome/a/answer/9296680>

Third-party products: This document describes how Google products work with the Microsoft Windows operating systems and the configurations that Google recommends. Google does not provide technical support for configuring third-party products. Google accepts no responsibility for third-party products. Please consult the product's website for the latest configuration and support information. You may also contact Google Solutions Providers for consulting services.

©2019 Google LLC All rights reserved. Google and the Google logo are registered trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated. [EXTENSIONS-en-1.0]

Introduction

Companies want to protect their user data. They want to easily vet extensions to be safe and relevant for their users. IT administrators need to:

1. Prevent bad extensions from being installed.
2. Keep extensions that users need.
3. Provide limited access to user and company data.

The goal of this guide is to show how you can manage extensions easily. There are multiple methods to manage extensions. This guide shows you the options and helps you pick the right method for you.

Considerations for managing Chrome extensions

Your users need access to certain apps, sites, and extensions to do their jobs. As the IT admin, you need to protect user and company data. A strategy is needed to choose how you will manage extensions.

Key questions to ask:

- What regulations and compliance measures do I need to follow?
- What kind of device or website access could go against my company's security policies?
- How much user or corporate data is stored on my user's machines?

As you make these decisions, Google provides policies that allow you to:

- Block or allow extensions based on your data protection policies.
- Force-install needed extensions on your users machines.
- Manage extensions while providing them with the least amount of rights needed to work.

The traditional way to manage is to allow or block specific extensions. There is an easier way. You can manage by the permissions that are needed by extensions. Research which permissions you want to allow. Then enforce policies that allow or block extensions that meet your requirements.

What are extension permissions?

Extensions can require rights to make changes on a machine or a web page to run properly. These rights are called permissions. Developers must list what rights and access their extensions require. There are two main categories, but many extensions have both:

- Site permissions may access the websites that your users visit.
Examples: Modify a webpage, access cookies, modify tabs
- Device permissions may access the machine where the browser is running.
Examples: Access to USB port / storage / viewing screen

How do extensions update?

Extensions only update when Chrome is running and the update occurs within the first few minutes of Chrome launching and then occurs again every 5 hours.

- Extensions are updated via this process:
 - a. Chrome sends a request that contains a list of installed extensions & versions to a Google server
 - b. Our servers respond with a set of instructions about which extensions to update.
 - c. Chrome then requests the CRX files for each of the out-of-date extensions and applies the update locally.
- How extensions can become out of date:
 - a. Due to large update download size or if the users has many extensions the update that might not complete during a short user session
 - b. Chrome not being launched
 - c. Extension developers have chosen to limit the amount of clients that they deploy an update to.
 - d. If an enterprise is self hosting an extension- this could be due to an access issue or configuration error
 - e. Other issues that would be attributed to errors in the development of the extension.

A resolution to out-of-date extensions is to either uninstall and reinstall an extension or you can manually force an update of an extension through `chrome://extension>enable developer mode>hit the update button`.

Managing extensions

Most organizations should manage extensions by their permissions and what websites they have access to. This method is more secure, easier to manage, and scalable.

This method saves you time as you only need to set the policies once. The days of managing long allow and block lists are gone. You can still include a small block list of extensions that should not be installed. And with the run-time hosts policy, your most important sites will be protected. To manage extensions in your organization with this method:

1. Find out which extensions are installed on your users' computers..
 - **Method 1(Recommended):** Use [Chrome Browser Cloud Management](#). This feature is offered at no additional cost for your users. You will be able to see the extension's:
 - Installed version, install count and if it was user or admin installed
 - Permissions required
 - Status (active or disabled)
 - The steps to setup Chrome Browser Cloud Management are located [here](#).

- Once you have the console setup and have your machines enrolled with cloud reporting enabled, you will be able to view all of the installed extensions under **Devices>Chrome>Apps and extensions usage report**
 - Clicking on an extension will provide you a bit more information on the permissions that it requires and examples of where it is installed
 - Coming later in late 2021/early 2022, when you click on a extension it will take you to the new extensions details page (pictured below)
 - You can get more insights about the extension including the permissions required and information directly from the Chrome Web Store listing
 - For more information on managing extensions in Chrome Browser Cloud Management, check out this [YouTube video](#).
 - You can also use the Takeout API from Chrome Browser Cloud Management to export all extension data from enrolled browsers into a CSV file.
 - For more information see : [Step by step guide](#) | [Blog entry](#) | [Demo Video](#)
 - **Method 2: Survey:** Ask your coworkers and their managers about what extensions they use regularly. Build a list of the extensions that users require.
2. Choose which sites you need to be secure:
 - Find out which sensitive websites or domains you need to block extensions from making changes or reading data.
 - You will prevent access to these sites by blocking the API calls when the extension is run. These include blocking web requests, reading cookies, JavaScript injection, XHR, etc.
 3. Identify which permissions could pose risks to your users:
 - Review the list of the extensions that you created in step one. Review the extensions that are installed and what permissions they require.
 - **Top Tip:** Permissions that extensions use can be vague. For your required extensions, reach out to the vendor to get more information. They should be able to detail the changes that the extension could make on machines and websites.
 - Examine the [Declare Permissions list](#). This lists all permissions an extension can use. Then decide which permissions you want to allow in your organization.
 - For more information about the risks of specific extensions permissions, review this document on [Permission Risks](#).
 4. Create a list from the data you collected, including:
 - **Required extensions:** This list could be broken down by department, office location, or other relevant information.
 - **Allow list:** Required extensions with permissions that would be blocked but need to be allowed to run. Examples could be:
 - Extensions needed by your users
 - Those determined to not be a risk through conversations with the vendor.

- **Block list:**
 - Extensions that will be blocked from installation.
 - This list includes the permissions that aren't allowed to run.
 - List the websites and domains that need to be kept secure and will not have extension access.
 - Compare this block list to others you have in place. You might find that you can relax your current block list policies.
- 5. Present your list to your stakeholders and IT team for approvals..
- 6. Test out the new policy in your lab or with a small pilot in your organization.
- 7. Roll out these new sets of policies to employees in phases.
- 8. Review feedback from your users.
- 9. Repeat and fine-tune the process monthly, quarterly, or yearly.

This will create a baseline of the permissions you allow and block others. Sensitive websites will be protected. Your browser security will improve with a better experience for users. Employees might be able to install extensions that they couldn't before. On your sensitive websites, they just won't run, unless you want them to. For steps on how to set up this method check out these sections in the guide:

- [Manage extensions through blocking /allowing permissions](#)
- [Runtime block hosts](#) (protect sensitive websites)
- [Force installing extensions](#) for your users
- [Allow/block \(if required\)](#) extensions

For an overview on managing extensions within Chrome Browser Cloud Management, check out this [YouTube video that covers extension management in the admin console](#).

Overview of the different extension management policies

Many of these policies will be covered in detail in the other sections of the document but here is a review of some of the options that you currently have today for managing extensions (some also apply to apps) via Windows Group Policy or via Plists on Macs:

- [Extension Install Allow List](#): These are the extensions that you have approved to be installed within your environment.
- [Extension Install Block List](#): These are the extensions that you will not allow to be installed. If they are installed already, they will be disabled. If a user tries to install them, it will be blocked. As well as there is a new feature within the Chrome web store where the add to Chrome button will be red and advise the user that the extension is not allowed to be installed.
- [Extension Install Force List](#): This will silently install the extension on your user's machine. The user can not disable or uninstall the extension. This setting will override the extension block list policy.
- [Block External Extensions](#): This setting will block extensions from external sources being installed. An example of this is if an installed application is adding an extension to Chrome via the registry, this setting will block that extension from loading.
- [Extension Allowed Types](#): Here you can create a list of what types of extensions and apps you will allow to be installed. Extensions, themes, user scripts, hosted applications, legacy packaged applications and platform applications are the values that are supported.
 - Note that whatever you want to allow must be included in the list. Anything left off the list will not be installed.

- For more information on the different types, here is a link on [Extensions and Apps in the Chrome web store](#).
- [Extension Install Sources](#): Previously users could click on a link to a .crx file and Chrome would offer to install the extension after a few warnings. This functionality was removed for security reasons after Chrome 21.
 - This policy allows you to get that older install functionality for specific URLs that you specify in this policy. Here is [a link to the URL match patterns](#) that can be used in this policy.
- [Extensions Settings](#): This policy provides a varied amount of functionality and requires a JSON script to be created and required to be formatted in a single line string.
 - This setting can be complex and will be covered in depth in various sections of this document.
 - It is recommended to consider using Chrome Browser Cloud Management as almost all of the functionality is included without the need of writing JSON as well as the ability to audit installed extensions.

A note on Google's commitment to inclusive naming conventions. The following policies have been deprecated and will be removed in Chrome 97, so make sure that you switch over to the new policy by then.

- [ExtensionInstallWhitelist](#) replaced with [ExtensionInstallAllowlist](#)
- [ExtensionInstallBlacklist](#) replaced with [ExtensionInstallBlocklist](#)


Block extensions based on their permissions

You can control extensions your users can install by their permissions. An installed extension with blocked permissions will be disabled. If a user tries to install one with a blocked permission, it won't install.

Manage extensions by their permissions in the Chrome Browser Cloud Management

(Windows, Mac and Linux)

You can block extensions that need permissions which aren't allowed. For example, you could block extensions from connecting to USB devices or prevent access to cookies.

1. In your Admin console, go to **Devices > Chrome > Apps and extensions > Users and browsers**
2. Select the organizational unit with the users you want to allow extensions for.
3. Click on the additional settings gear  **ADDITIONAL SETTINGS**

4. Check each permission to block or allow under the **Permissions and URLs section**.

Permissions and URLs
Locally applied ▾

Block extensions by permission

<input type="checkbox"/> Alarms	<input type="checkbox"/> Audio capture	<input type="checkbox"/> Certificate provider
<input type="checkbox"/> Clipboard read	<input type="checkbox"/> Clipboard write	<input type="checkbox"/> Context menus
<input type="checkbox"/> Desktop capture	<input type="checkbox"/> Document scan	<input type="checkbox"/> Enterprise device attributes
<input type="checkbox"/> Experimental APIs	<input type="checkbox"/> Fullscreen apps	<input type="checkbox"/> File browser handler
<input type="checkbox"/> File system	<input type="checkbox"/> File system provider	<input type="checkbox"/> HID
<input type="checkbox"/> Override fullscreen escape	<input type="checkbox"/> Detect idle	<input type="checkbox"/> Identity
<input type="checkbox"/> Google Cloud Messaging	<input type="checkbox"/> Geo location	<input type="checkbox"/> Media galleries
<input type="checkbox"/> Native messaging	<input type="checkbox"/> Captive portal authenticator	<input type="checkbox"/> Power
<input type="checkbox"/> Notifications	<input type="checkbox"/> Printers	<input type="checkbox"/> Serial
<input type="checkbox"/> Set proxy	<input type="checkbox"/> Platform keys	<input type="checkbox"/> Storage
<input type="checkbox"/> Sync file system	<input type="checkbox"/> CPU metadata	<input type="checkbox"/> Memory metadata
<input type="checkbox"/> Network metadata	<input type="checkbox"/> Display metadata	<input type="checkbox"/> Storage metadata
<input type="checkbox"/> Text to speech	<input type="checkbox"/> Unlimited storage	<input type="checkbox"/> USB
<input type="checkbox"/> Video capture	<input type="checkbox"/> VPN provider	<input type="checkbox"/> Web requests
<input type="checkbox"/> Block web requests		

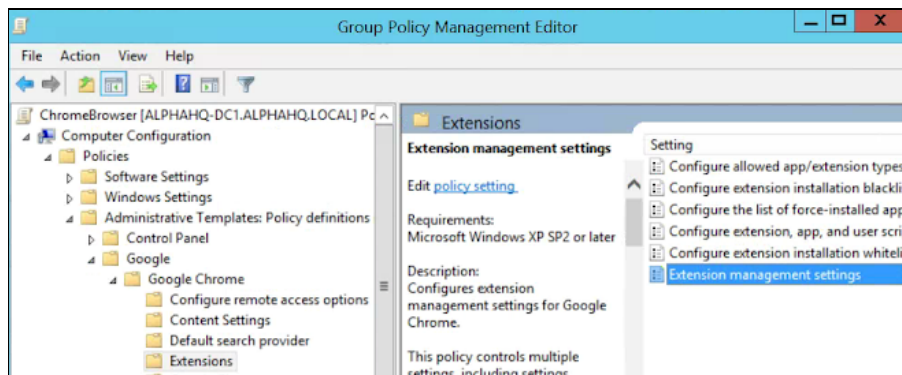
- a. You can also click on an individual extension in the Users and browsers tab and manage via permissions under Permission and URL Access> Customize Permissions for this app/extension.
 - i. Note: that this will override any global policy that is already applying to this extension.
 - ii. For complete details on each permission, see this [list of permissions](#).

5. Click **Save**.

Manage extensions by their permissions in Group Policy

(Windows Only)

1. Browse to the Group Policy object in the Microsoft Management console.
2. Right-click > Click **Edit**.
3. In the group policy management editor, browse to **Policies > Administrative Templates > Google Chrome > Extensions > Extensions management settings**.



Configure extension management settings path

4. Enable the policy, then enter the permissions that you want allowed or blocked, compressing it to a single JSON string.

Format according to this example JSON data. (This example blocks any extension that needs use of USB.)

```
{
  "*": {
    "blocked_permissions": ["usb"]
  }
}
```

Compact JSON data:

```
{"*":{"blocked_permissions":["usb"]}}
```

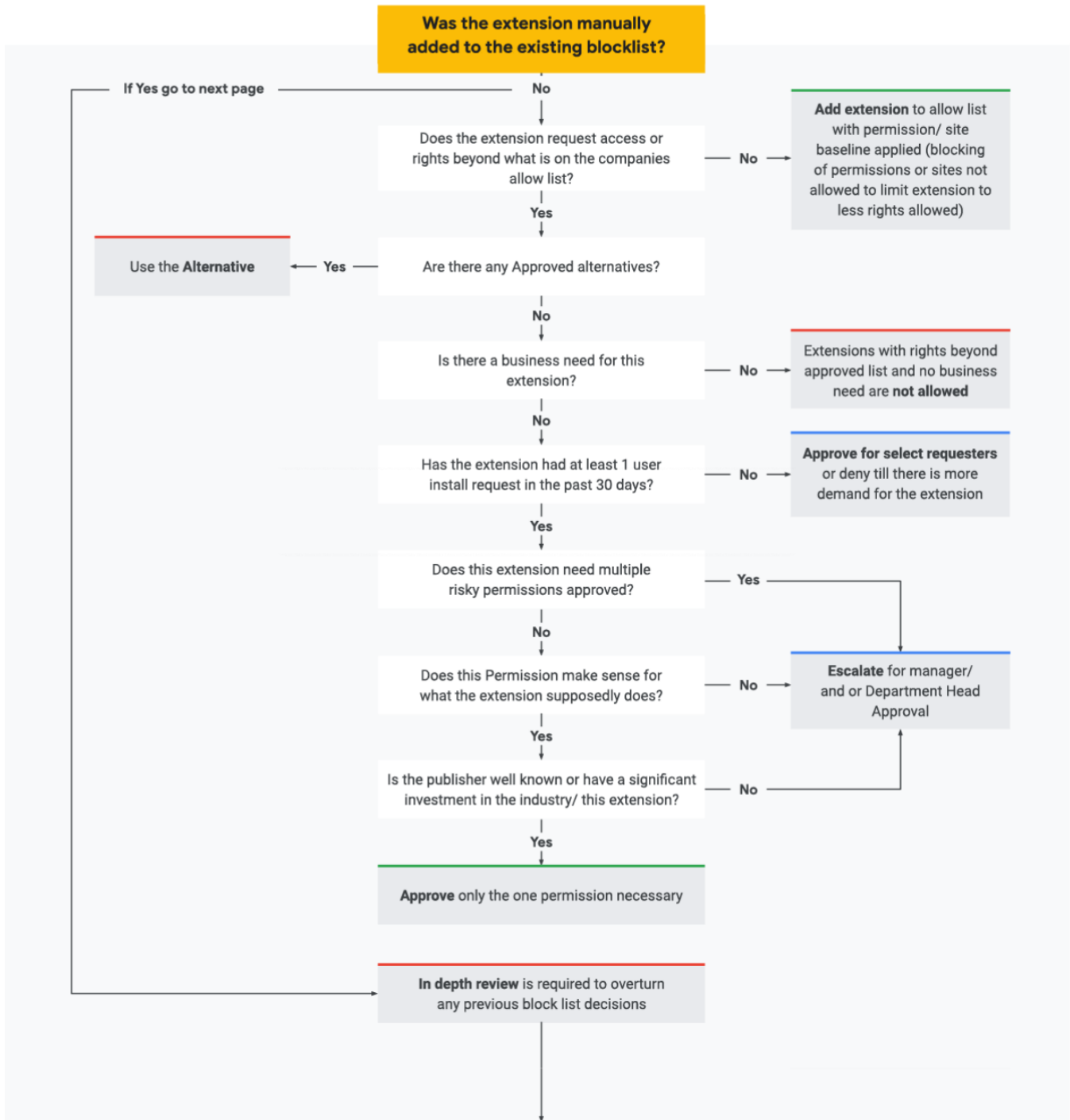
Top Tip:

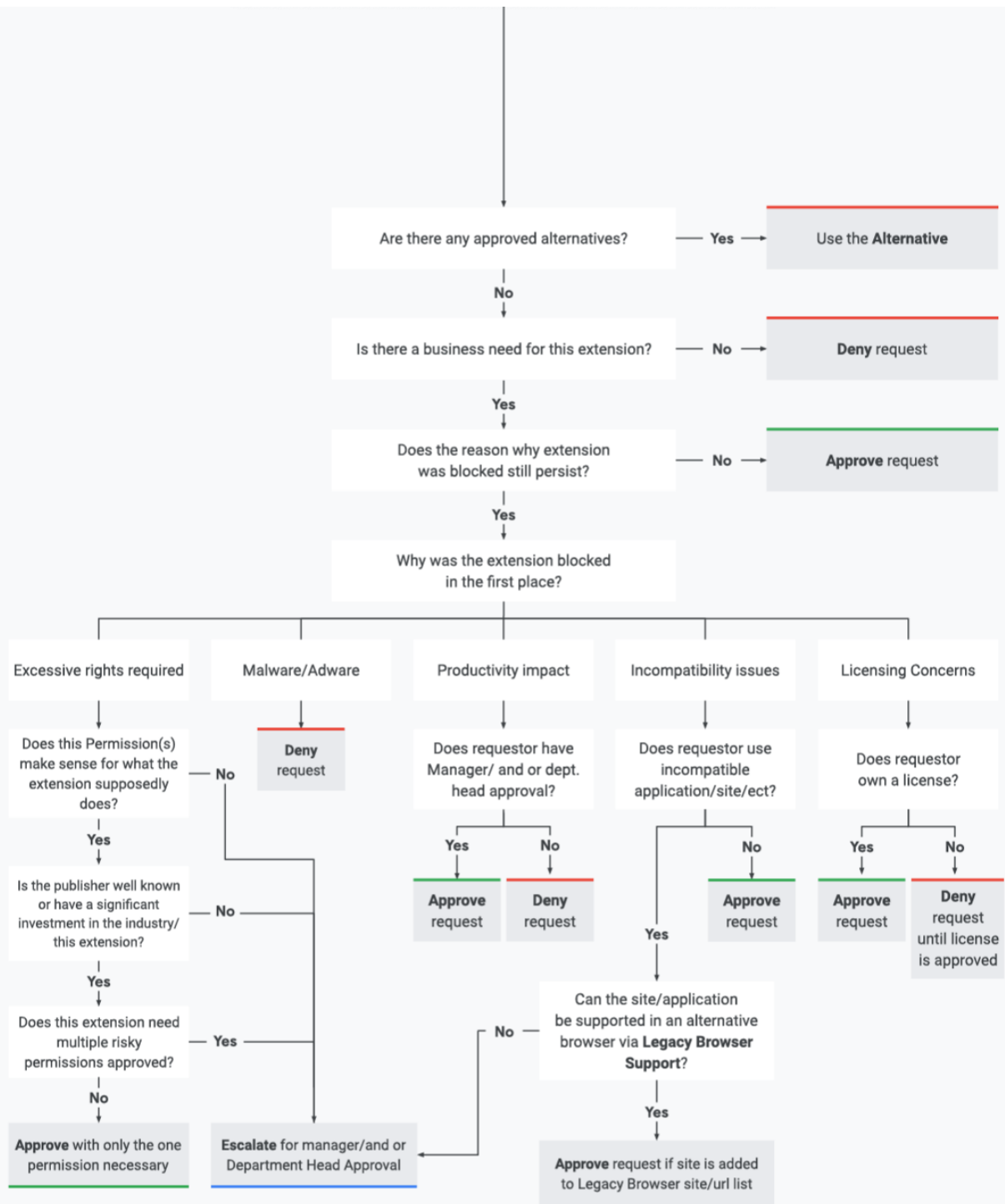
- To block all extensions that use that permission, use an asterisk (as shown above) for the extension ID.
- If you want to block multiple permissions via JSON here is an example that blocks power, printerProvider, serial and usb for all extensions:
 - `{"*":{"blocked_permissions":["power","printerProvider","serial","usb"]}}`
- If you specify one extension ID, the policy will only apply to that extension. In the example above, replace the * with the extension ID. You can block more than one, but they need to be separated into their own entries in the JSON string.
 - For steps for finding the extension ID , see step 3 of [this help article](#).

Creating an exception process for extensions that require risky permissions

There might be a business need for extensions that might require permissions that you have deemed to be too risky to be run in your environment. To give you an idea of what an exception workflow might look like, here is an example workflow for a requested extension that requires a currently blocked extension.

Start here





Note that this flow is just provided as an example as each enterprise will have their own workflow or change management processes.

Managing extensions by the extensions setting policy

Windows offers multiple ways to manage extensions. A common way is to set multiple policies with a JSON string or in the Windows Registry using the [extensions settings policy](#).

Top Tip: This policy is supported on [Mac](#), [Chrome OS](#) and [Linux](#). [The policy page](#) has example values for these other platforms.

This policy can control settings such as the update URL, where the extension will be downloaded from for initial installation, and blocked permissions, which are not allowed to run. For more info, read the [Extension settings full description](#). There is also further information in these help articles: [Configure ExtensionSettings policy](#) and [App and extension policies](#).

You can decide if you want to set all extension management settings via this policy or through individual policies.

- The runtime allowed/blocked hosts setting (blocking extensions on specific websites) can only be set via GPO within the extension settings policy.
 - It can also be set via the [Chrome Browser Cloud Management](#).
- Note that the extension settings policy can overwrite other policies that you have elsewhere in group policy, including:
 - [ExtensionAllowedTypes](#)
 - [ExtensionInstallAllowlist](#)
 - [ExtensionInstallForcelist](#)
 - [ExtensionInstallSources](#)
 - [ExtensionInstallBlocklist](#)

The extension settings policy is set by one of these 2 methods:

- [Windows Registry](#)
- [JSON string in Windows Group Policy Editor](#)

Top Tips:

- Correctly formatting a JSON string can be tricky. Use a JSON checker before implementing the policy.
- If you struggle with getting the JSON formatted correctly, you can use the registry key method and Chrome will convert it to JSON within `chrome://policy` within the browser on the target machine.
 - Just copy that JSON and you can apply it via GPO via the extension settings policy.
 - You can also use this method through setting extension settings via Chrome Browser Cloud Management and copying the JSON output.

Configure the extension policy using the Windows Registry

The ExtensionSettings policy needs to be written to the registry under:

HKLM\Software\Policies\Google\Chrome\ExtensionSettings\

- It's possible to use HKCU instead of HKLM. The equivalent path can be configured with GPO.
- The keys can be created with your chosen method on your user's machine.

For Chrome, all settings will start under this key:

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\ExtensionSettings\

The next key that you will create is for the scope of the policy. Name the key after the extension ID for applying to one extension. Name the key with an asterisk to apply to all extensions. For example, use the following location for settings that apply just to the Google Hangouts extension:

HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\ExtensionSettings\nckgahadag
oajjgafhacjanaoiihapd

For settings that apply to all extensions, use this location:



HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\ExtensionSettings*

Different settings will require different formats, depending on whether they are a string or an array of strings. Array values require [" value "]. String values can be entered without the [" "]. The list of which settings are arrays or strings:

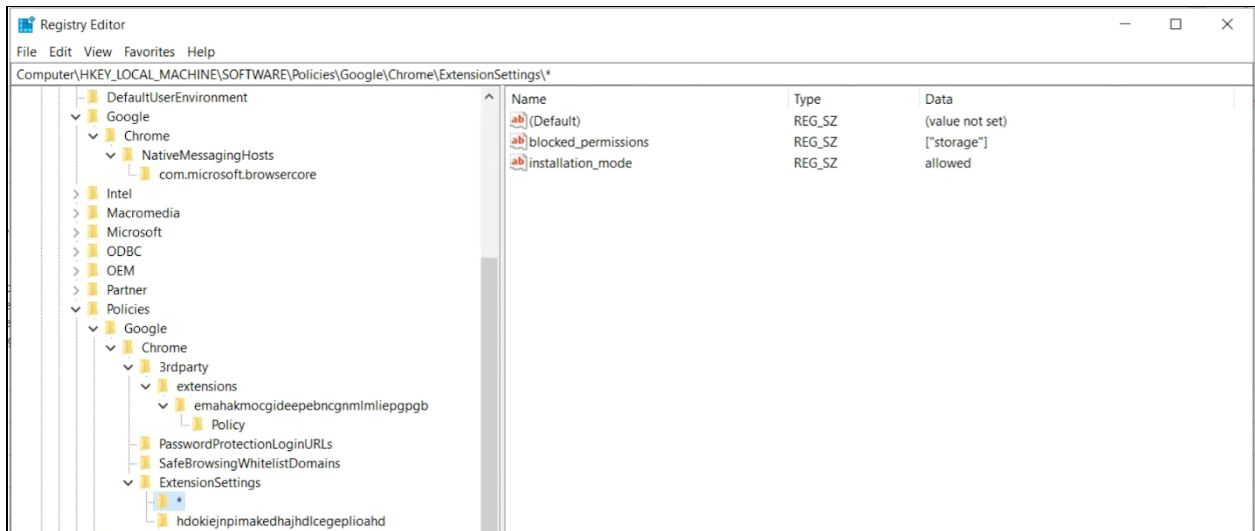
- Installation_mode = String
- update_url = String
- blocked_permissions = Array of strings
- allowed_permissions = Array of Strings
- minimum_version_required = String
- runtime_blocked_hosts = Array of strings
- runtime_allowed_hosts = Array of Strings
- blocked_install_message = String

If you want to set multiple values in a single string (like blocked permissions) here is an example of that syntax:

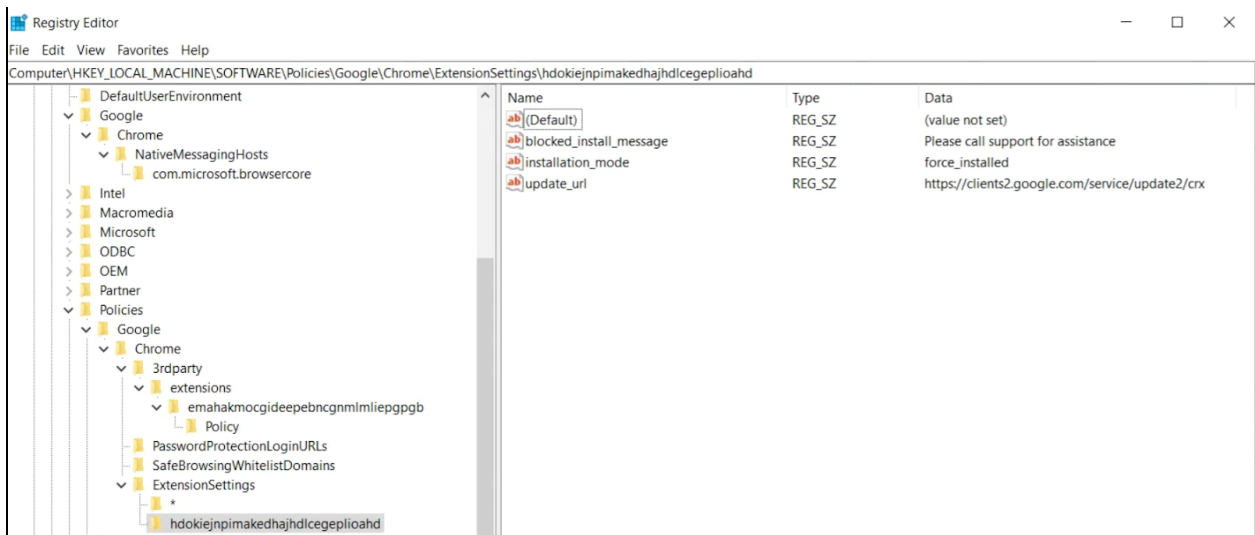
- ["power","printerProvider","serial","usb"]

Name	Type	Data
 (Default)	REG_SZ	(value not set)
 blocked_permissions	REG_SZ	["power", "printerProvider", "serial", "usb"]

Examples of what the keys look like within the registry:



The default (*) scope key and its values



An individual scope and its values

Here, the keys set in the registry are converted to JSON with the policy in chrome://policy within the browser:

Chrome policies

Applies to	Level	Source	Policy name
Machine	Mandatory	Platform	DefaultBrowserSettingEnabled
Machine	Mandatory	Platform	ExtensionSettings

```

{
  "*": {
    "blocked_permissions": [ "storage" ],
    "installation_mode": "allowed"
  },
  "hdokiejnpimakedhajhdlcegeplioahd": {
    "blocked_install_message": "Please call support for assistance",
    "installation_mode": "force_installed",
    "update_url": "https://clients2.google.com/service/update2/crx"
  }
}

```

Configure using a JSON string in Windows Group Policy Editor

The steps to use the extension settings policy using GPO assume you have already imported the [ADM/ADMX for Chrome Policies](#).

For other OS platforms, check: [Mac](#) | [Linux](#) | [Chrome OS](#)

1. Within the GPO management editor, go to **Google Chrome > Extensions > Extensions management setting policy**.
2. Enable the policy and enter its compact JavaScript Object Notation (JSON) data in the text box as a single line with no line breaks.
To validate policies and compact them into a single line (example JSON data below), use this [third-party JSON compression tool](#).

Properly formatting JSON for the extension settings policy:

To use this method, you need to understand the 2 parts to this policy—the **default** and the **individual** scope. The default scope applies to all extensions. The individual scope is applied to the specified extension only.

The default scope is identified by the asterisk (*). This example defines a default scope and a single individual extension scope:

```

{
  "*": {},
  "nckgahadagoaajjgafhacjanaoiihapd": {}
}

```

An extension will only get its settings from one scope. If there's an individual scope for that extension, those settings will apply. If no individual extension scope exists, it will use the default scope.

Here is an example JSON that blocks any extension from running on .example.com and blocks any extension that requires the permission "USB":

```
{
  "*": {
    "runtime_blocked_hosts": ["*://*.example.com"],
    "blocked_permissions": ["usb"]
  }
}
```

Compact JSON data:

```
{"*":{"runtime_blocked_hosts":["*://*.example.com"],"blocked_permissions":["usb"]}}
```

Reference examples with example values for installation management of extensions:

- "allowed" (default)
Your user can install the extension from the Chrome Web Store.
Example JSON:

```
{ "*": {"installation_mode": "allowed" } }
```
- "blocked"
Your user can't install the extension from the Chrome Web Store.
Example JSON:

```
{ "*": {"installation_mode": "blocked" } }
```
- "blocked_install_message"
Here you can specify a custom message to display when installation is blocked.
Example JSON - blocked_install_message:

```
{ "*": {"blocked_install_message": ["Call IT(408 - 555 - 1234) for an exception"] } }
```
- "force_installed"
 - The extension is automatically installed without your user's interaction.
 - Your user can't disable or remove the extension.

```
{ "*": {"installation_mode": "force_installed" } }
```
- "normal_installed"
The extension is automatically installed without your user interaction, but they can disable the extension.

```
{ "*": {"installation_mode": "normal_installed" } }
```
- "removed"
(Chrome version 75 or later) Users can't install the extension. If users previously installed the extension, Chrome Browser removes it.

```
{ "*": {"installation_mode": "removed" } }
```

- "toolbar_pin"

Controls if the extension icon is pinned to the toolbar. You can set it to:

force_pinned—The extension icon is pinned to the toolbar and visible at all times. The user can't hide it in the extension menu.

default_unpinned—The extension starts hidden in the extension menu, and the user can pin it to the toolbar.

If you do not set this field, it defaults to the default_unpinned behavior.

```
{ "*" : { "toolbar_pin" : "forced_pinned" } }
```

If an extension uses the installation_mode feature then another field "update_url" must also be defined, pointing to where the extension can be installed from.

- If the extension you're downloading is hosted on the Chrome Web Store, use ["https://clients2.google.com/service/update2/crx"](https://clients2.google.com/service/update2/crx).
- If you're hosting the extension on your own server, put the URL where Chrome can download the packed extension (.crx file).
Example JSON - force_installed extension with update_url:

```
{ "nckgahadagoaajjgafhacjanaoiihapd" : { "installation_mode" :  
"force_installed", "update_url" :  
"https://clients2.google.com/service/update2/crx" } }
```
- Since Chrome 89 you can also use the override_update_url setting to specify that Chrome uses the URL in the update_url field or the update URL specified in the ExtensionInstallForcelist policy for subsequent extension updates.
 - If this is not set or is set to false, Chrome uses the URL specified in the extension's manifest for updates instead.

Prevent extensions from altering webpages

This setting prevents extensions from changing and reading data from your most sensitive websites.

This policy will block extensions from:

- Injecting scripts into your websites
- Reading the cookies
- Making web-request modifications

This setting doesn't prevent users from installing or removing extensions. It only prevents extensions from altering websites that you specify.


There are two settings you can use for this feature

- **runtime_blocked_hosts** - Extensions are blocked from interacting with these hosts
- **runtime_allowed_hosts** - Extensions may interact with hosts on this list even if defined in runtime_blocked_hosts.

Top Tip: Each instance of runtime_blocked_hosts and runtime_allowed_hosts can have at most 100 Host Patterns. If you define more than that, your policy will be invalid.

Chrome Browser Cloud Management

Blocking by runtime host is simpler within [Chrome Browser Cloud Management](#) than in GPO. It requires no JSON and is as simple as entering the URL that you want to block in the extension settings. To set this up, you need to enroll your browser devices into Chrome Browser Cloud Management. The feature is offered at no additional cost. The steps for enrollment are [located here](#).

1. In your Admin console, go to **Devices > Chrome > Apps and Extensions > Users and browsers**
2. Select the organizational unit with the users you want to allow extensions for.
3. Click on the additional settings gear 
4. Enter the URL of the sensitive websites that you do not want the extensions to run on in the "runtime blocked hosts" section. For syntax info review [syntax for blocked or allowed Urls](#)
 - a. You can enter multiple URLs by hitting enter after each URL for a new entry.
 - b. You can also click on an individual extension and set allowed and blocked hosts in the permissions and URL access section.
 - i. Note: that this will override any global policy that is already applying to this extension.
 - ii. There is also an allowed_hosts section for exceptions for URLs that are listed in the runtime block hosts section.
5. Click **Save**.

Runtime blocked hosts

://.sensitivesite.com

This is a list of patterns for matching against hostnames. URLs that match one of these patterns cannot be modified by apps and extensions. This includes injecting Javascript, altering and viewing webRequests / webNavigation, viewing and altering cookies, exceptions to the same-origin policy, etc.
The format is similar to full URL patterns except no paths may be defined. e.g.
"*://*.example.com"

Runtime allowed hosts

Hosts that an extension can interact with regardless of whether they are listed in "Runtime blocked hosts".
This is the same format as "Runtime blocked hosts".

Runtime hosts section in **Devices > Chrome > Apps and Extensions > Users and browsers > Additional settings**

GPO

These instructions are for managing this GPO on Windows machines. For other platforms, check: [Mac](#) | [Linux](#)

Within the Extension Settings policy, you can set the following settings to block (or allow) alterations of websites or domains:

- `Runtime_blocked_hosts`
This setting blocks extensions from making changes or reading data from your chosen websites.
- `Runtime_allowed_hosts`
This setting allows extensions to make changes or read data from your chosen websites.

The format for specifying your site(s) in the JSON string in either policy is:

```
[http|https|ftp|*]://[subdomain|*].[hostname|*].[eTLD|*] [http|https|ftp|*],
```

Note: `[hostname|*]`, and `[eTLD|*]` sections are required, but `[subdomain|*]` section is optional.

Examples of valid host patterns and matching patterns:

Valid host patterns	Matches	Doesn't match
<code>*://*.example.*</code>	<pre>http://example.com https://test.example.co.uk</pre>	<pre>https://example.google.com http://example.google.co.uk</pre>
<code>http://example.*</code>	<pre>http://example.com http://example.ly</pre>	<pre>https://example.com http://test.example.com</pre>
<code>http://example.com</code>	<pre>http://example.com</pre>	<pre>https://example.com http://test.example.co.uk</pre>
<code>*://*</code>	All Urls	

Here is a sample of a JSON string that blocks access for a single extension. This string prevents an single extension from augmenting a specific site:

```
{
  "aapbdbdomjkkjkaonfhkkikfgjllcleb": {
    "runtime_blocked_hosts": ["*://*.importantwebsite"]
  }
}
```

Compact JSON data:

```
{"aapbdbdomjkkjkaonfhkkikfgjllcleb":  
{"runtime_blocked_hosts":["*://*.importantwebsite"]}}
```

Here is a sample for blocking multiple sites for all extensions:

```
{  
  "*": {"runtime_blocked_hosts": [ "*://*.importantwebsite.com",  
    "*://*.importantwebsite2.com" ]  
}
```

Compact JSON data:

```
{"*":{"runtime_blocked_hosts":["*://*.importantwebsite.com","*://*.importantweb  
site2.com"]}}
```

For multiple extensions, separate each into its own entry for each app ID that you want to block. Here's an example of how to block two extensions from running on the same domain:

```
{  
  "aapbdbdomjkkjkaonfhkkikfgjllcleb": {  
    "runtime_blocked_hosts": ["*://*.importantwebsite"]  
  },  
  "bfbmjmiodbnnpllbbbfblcplfjjepjdn": {  
    "runtime_blocked_hosts": ["*://*.importantwebsite"]  
  }  
}
```

Compact JSON data:

```
{"aapbdbdomjkkjkaonfhkkikfgjllcleb": {"runtime_blocked_hosts":  
["*://*.importantwebsite"]}, "bfbmjmiodbnnpllbbbfblcplfjjepjdn":  
{"runtime_blocked_hosts": ["*://*.importantwebsite"]}}
```

Allow or block extensions in the Google Admin console

Admins can control which extensions your users can install by creating allow and block lists. You can allow users to install any app or extension. You can set policies to block or allow apps for all users or certain employees.

The following steps assume you're familiar with changing settings in your Admin console.

Allow all extensions except those you want to block

1. In your Admin console, go to **Devices > Chrome > Apps and extensions>Users and browsers>Additional settings**.
2. Select the organizational unit that you want to allow extensions for, on the left.
3. Scroll down to the Allow/block mode section under Chrome Web Store, click edit and select the option to **Allow all apps, admin manages blocklist**.

Edit Allow/block mode setting

Play Store

Allow all apps, admin manages blocklist ▼

Chrome Web Store

Allow all apps, admin manages blocklist

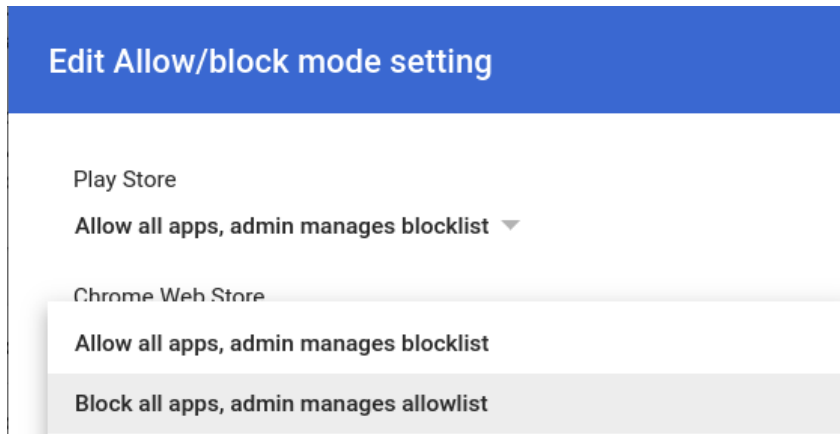
Block all apps, admin manages allowlist

Allow block mode setting

4. Click **Save**
5. Click on the Users and browsers tab to return to the previous page.
6. Add each extension you want to block by clicking on the yellow plus mark in the bottom right.
7. Choose your method of adding it into the console (add from Chrome Web store, Add by extension ID, Add by URL)
8. Select the dropdown by the extension and select **block**.
9. Click **Save**.

Block all extensions except those you want to allow

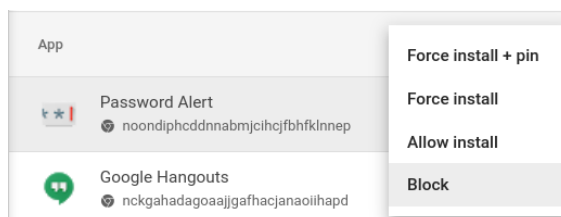
1. In your Admin console, go to **Devices > Chrome> Apps and extensions>Users and browsers>Additional settings**.
2. Select the organizational unit that you want to block extensions for, on the left.
3. Scroll down to the Allow/block mode section under Chrome Web Store, select the option to **Block all apps, admin manages allowlist**.



4. Click **Save**
5. Click on the Users and browsers tab to return to the previous page.
6. Add each extension you want to allow by clicking on the yellow plus mark in the bottom right.
7. Choose your method of adding it into the console (add from Chrome Web store, Add by extension ID, Add by URL)
8. Select the dropdown by the extension and select **allow install**.
 - a. You can also force install the extension to your user machines by selecting Force install.
9. Click **Save**.

Block or allow one extension

1. In your Admin console, go to **Devices > Chrome> Apps and extensions>Users and browsers**
2. Select the organizational unit that you want to allow or block the extension for.
 - o Note: that the organizational unit will inherit the settings of the parent Organizational unit, but you can override per sub organizational unit.
3. Select the extension you want to block or allow or add it in (see steps 6 & 7 of the previous section).
4. In the installation policy column select block, force Install or allow install.

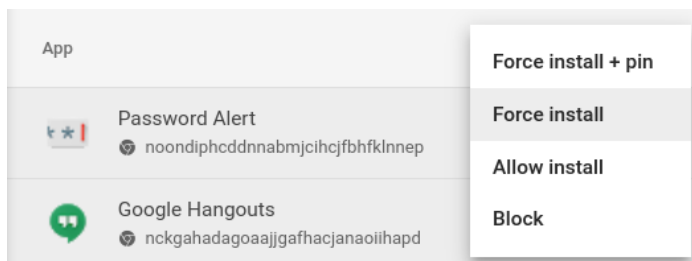


5. Click **Save**.

Force-installing extensions

If you know that the user requires an extension, you can install it for them. If you force install an extension, it will grant all the permissions it needs to run. The user also will not be able to remove it and it will be installed silently. If you remove an extension from the force install list, it will be removed from the user's machine.

1. In your Admin console, go to **Devices > Chrome> Apps and extensions>Users and browsers**
2. Select the organizational unit that you want to force install extensions to.
3. Select the existing extension(s) you want to force install or add them in.
 - a. To add the extension(s) you want to install, click on the yellow plus mark in the bottom right.
 - b. Choose your method of adding it into the console (add from Chrome Web store, Add by extension ID, Add by URL)
4. Select the extension(s) that you want to force install and in the installation policy column, select **Force install** from the dropdown menu.



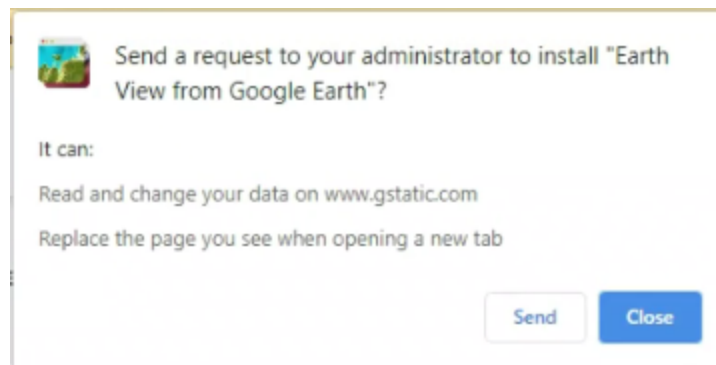
5. Click **Save**.

Note that you can create a custom Chrome Web Store collection of admin selected extensions that will be displayed to your users. This setting does require your users to be signed into Google identity using corporate credentials.

- This setting can be found within the admin console under **Devices>Chrome>Apps and extensions>Users and browsers>Additional settings>Chrome Web Store homepage>Use the Chrome Web Store collection**
 - Then you can either have all of your extensions show up within this page or you can click on individual extensions under the users & browsers section and select the include in Chrome Web Store collection toggle.

Let users request Extensions: Extension Workflows

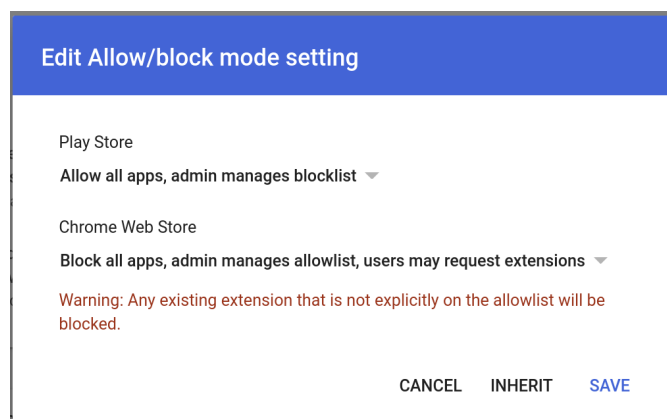
As an admin, you can use the Google Admin console to let users request the extensions that they need in the Chrome Web Store. Then, you can allow, block, or automatically install extensions that users request.



Example of request dialog from Chrome Web store

Note that this feature works as an allow/block list. When the feature is turned on, **all** extensions are blocked by default. To prevent any issues, it is recommended to follow this process:

1. Discover what extensions your users are currently using, through using the [extension takeout report](#) in Chrome Browser Cloud Management.
 - For more information check out this [Youtube video on setting up the Takeout API](#).
2. Create a list of essential extensions ([GPO](#) or [admin console](#)) based on the data gathered in step one.
3. Turn on the extension workflow feature under **Devices > Chrome > Apps and extensions>Users and browsers>Additional settings>Allow/block mode** and hit the edit button
4. Under Chrome Web store, select **“Block all apps,admin manages allowlist, users may request extensions”** from the drop down menu.



Enabling Extension Workflows in the admin console

- We recommend that first you apply settings to a small number of users and devices in a test organizational unit to prevent end user issues and gather feedback. Once you feel ready, then you can apply it to your entire organization.
- 5. Approval and denial requests are managed under **Devices>Chrome>Apps & Extensions>Requests**
- 6. Click the row of the extension request that you want to review..
- 7. Here you can review details about the extension and select the installation policy from the dropdown menu:
 - Force install- installs the extension silently and it can not be removed
 - Allow install – Lets users install the extension
 - Block – Prevents users from installing the extension. Removes the extension from users that have it installed

For more information about this feature, please review [the help center article for Extension Workflows](#) or this [Youtube video on extension workflow](#).

Allow or block extensions in Group Policy

Before you begin: The following steps assume that you already have Chrome managed for your users. For more on how to deploy Chrome on Windows, refer to [Chrome browser Deployment Guide \(Windows\)](#). For Mac[®] deployment and policy management, go to [Set up Chrome browser on Mac](#).

For Windows, there are two types of policy templates: an ADM and ADMX template. Ensure that you verify which type you can use on your network. The templates show which registry keys you can set to configure Chrome and what the acceptable values are. Chrome looks at the values set in these registry keys to determine how to act.

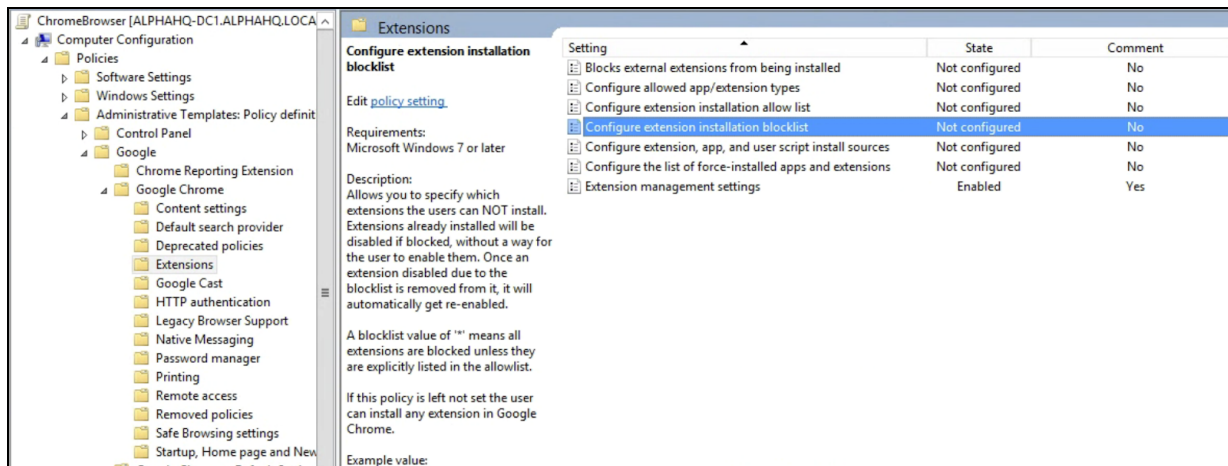
1. Download Chrome policy templates.
Windows templates, as well as common policy documentation for all operating systems, can be found in this [via this link](#)
2. Open the ADM or ADMX template you downloaded:
 - a. Go to **Start > Run: gpedit.msc**.
 - b. Go to **Local Computer Policy > Computer Configuration > Administrative Templates**.
 - c. Right-click **Administrative Templates** and select **Add/Remove Templates**.
 - d. Add the chrome.adm template through the dialog.

Afterward, if it's not already there, a Google or Google Chrome folder will appear under Administrative Templates.

- If you add the ADM template on Windows 7 or 10, it will appear under Classic Administrative Templates / Google / Google Chrome.

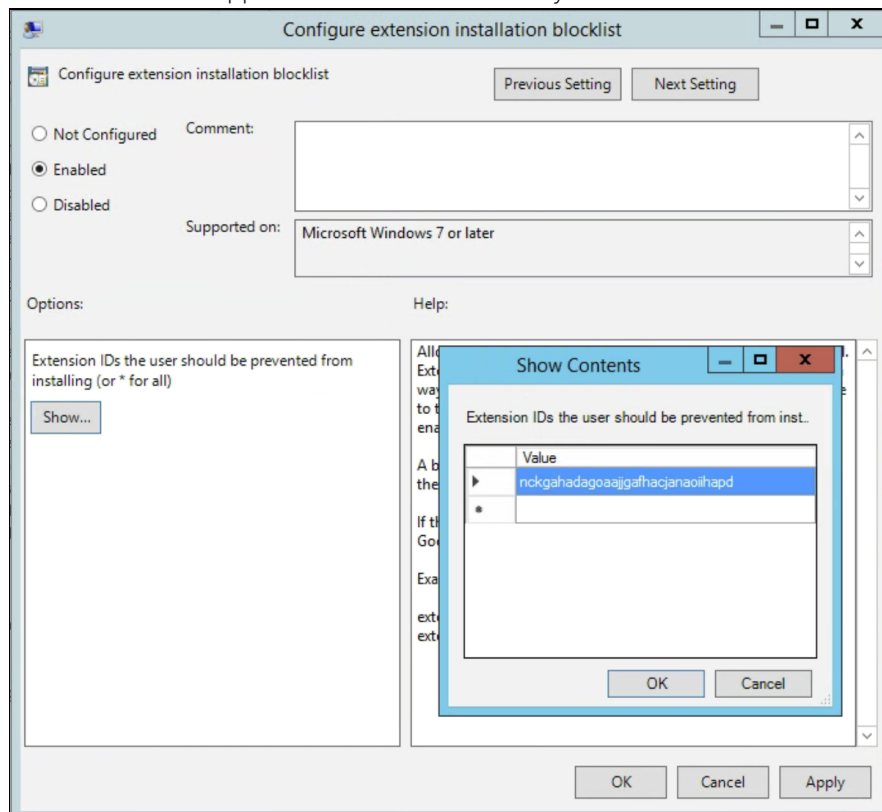
Allow all extensions except those you want to block

1. In the Group Policy Editor, open the template you just added.
2. Browse to **Google > Google Chrome > Extensions > Configure Extension installation blacklist**.



Path to Extension management policies

2. In the setting, select **Enabled**.
3. Click **Show**.
4. Enter the app ID of the extensions that you want to block.



Configure extension installation blacklist

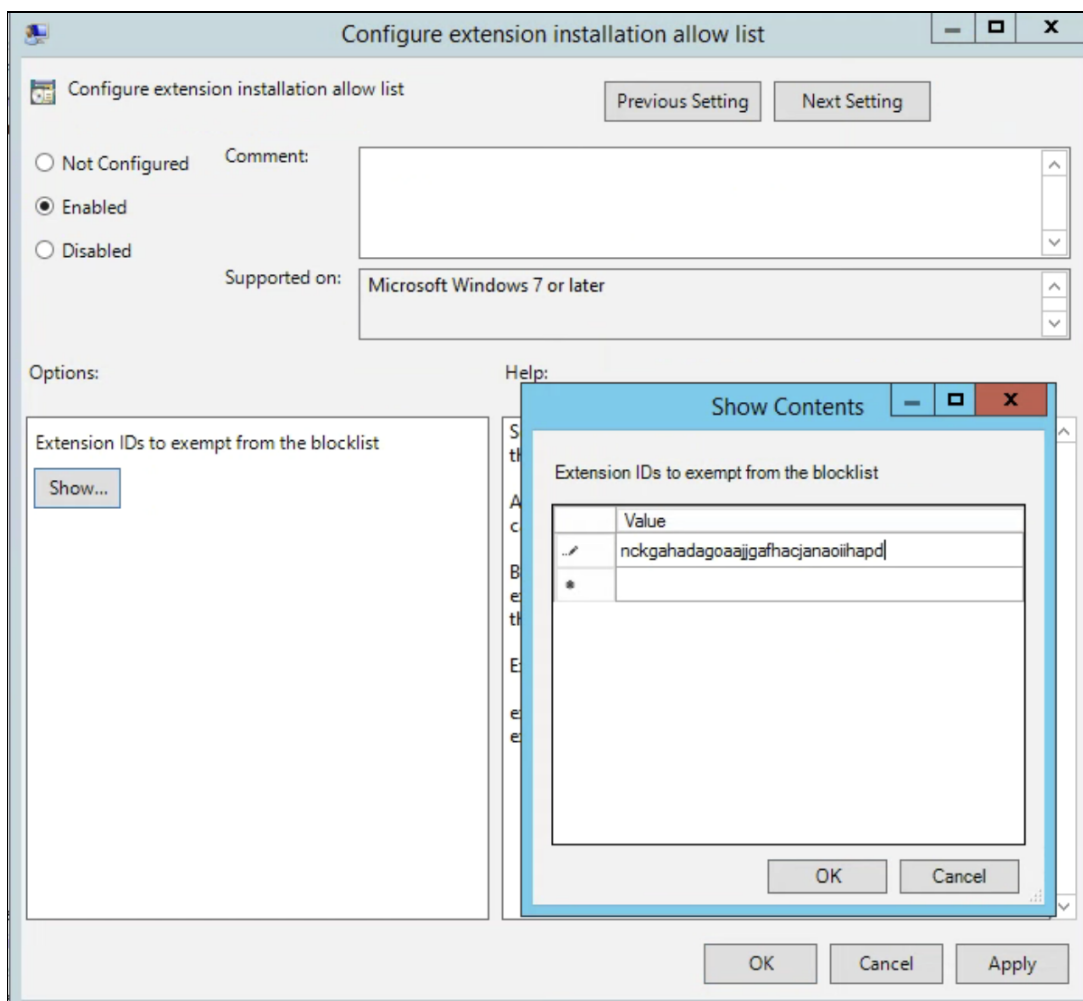
Notes:

- If you can't find the app ID of an extension, view it in the Chrome Web Store. Find the specific extension and the app ID will be at the end of the URL in the Chrome omnibox:



App ID example located after google-hangouts/

- Enter * into the policy to prevent any extensions from being installed. You can use this with the Configure extension installation allowlist policy. This way you only allow certain extensions to be installed by your users and block the rest.
- You can add an extension to the block list that is already installed on a user's machine. It will disable the extension and prevent the user from re-enabling it. It will not be uninstalled, just disabled.



Configure extension installation allowlist

Block or allow one extension

To block a single extension, add the app ID of the extension you want blocked to the configure extension installation blacklist policy. All of your other extensions will be allowed to be installed.

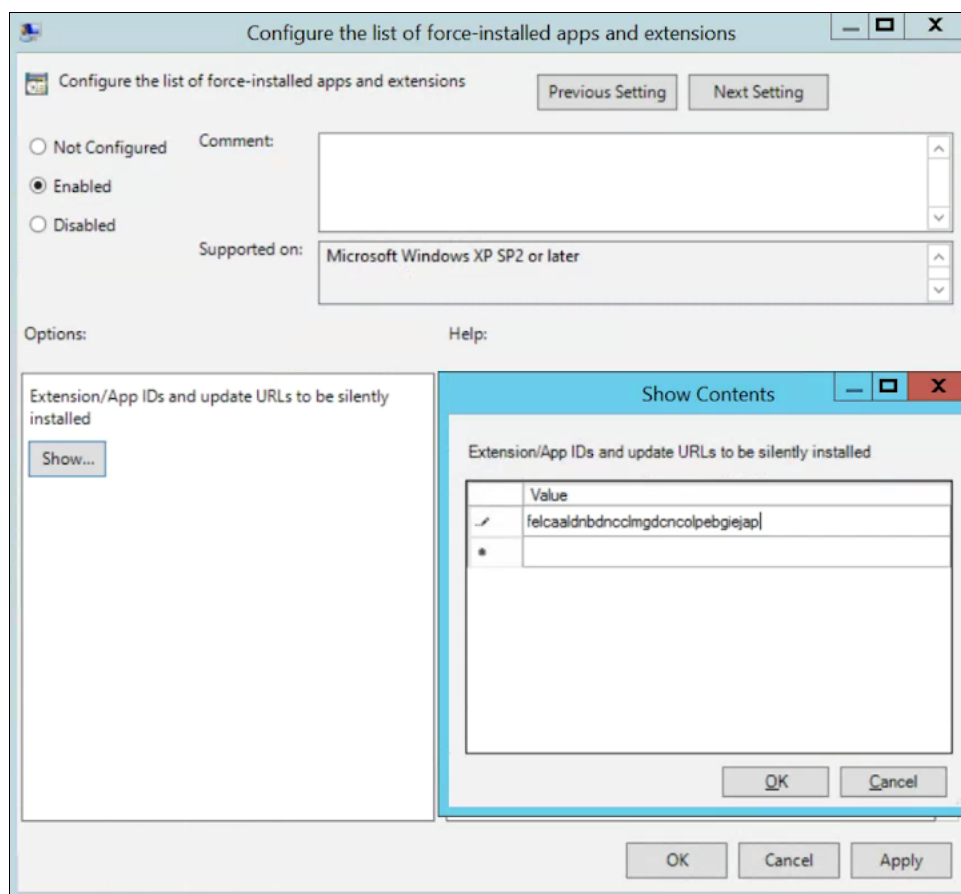
To allow only one extension:

1. In the content section in the Configure extension installation blacklist policy, enter *. This will block all extensions from being installed that are on the list.
2. Add the app ID of the allowed extension to the Configure extension installation allowlist policy.

Force-install an extension

1. In the Group Policy Editor, browse to **Google > Google Chrome > Extensions > Configure the list of force-installed apps and extensions**.
2. Select **Enabled**.
3. Click **Show**.
4. Enter the app ID or IDs of the extension or extensions you want to force-install.

The extension will be installed silently with no need for a user to interact. The user also won't be able to uninstall or disable the extension. This setting will override any block list policy that you might have enabled.



Configure the list of force-installed apps and extensions

Validating your Policy

To make sure your policy is valid and works as expected, apply it to a test machine. From the test machine follow these steps.

1. Browse to `chrome://policy`
2. Click the “Reload Policies” button
3. In the top right hand corner of the page is the policy filter, type in “ExtensionSettings” to only show this policy.
4. Check the “Show policies with no value set” checkbox
5. Make sure the “Status” for your policy shows “OK”
6. Expand the policy by clicking “Show Value” and make sure it isn’t empty
7. Congrats! You’ve got a valid policy

Self hosting your own extensions

The [Chrome Web Store](#) hosts extensions and provides many security features.

- Features such as automated and manual code scans.
 - This prevents malicious code from being sent to your users.

However there’s an option to host your extensions on your own server separate from the Chrome Web Store. Here are some of the pros and cons of this method:

Pros:

- Hosting your own extensions means that you are outside of the rules and requirements of the Chrome Web Store.
 - So there is less scrutiny on the extension and limits the risk that the extension could be removed for violating the terms of service.

Cons:

- The self-hosting method requires more setup and requires you to host your own file server for extension files.
- Validating the security of extensions and keeping them updated can be tough, where the Chrome Web Store does this automatically.

If you choose to self-host your extensions, this section tells you how. It covers how to package an extension and host it without using the Chrome Web Store. It also includes instructions on how to deploy these extensions to your devices and users.

Alternatives to self hosting extensions

Extension publishing options

As an alternative to self hosting, consider marking internal extensions on the Chrome Web Store as private. There are three options to publish extensions, public, private and unlisted. Here is a chart with more information about the pro and cons of each:

	Present in Chrome Web Store Search?	Require Sign-in?	Supported in Chrome Browser Cloud Management
Public	Yes	No	Yes
Private	No	Yes	Yes
Unlisted	No	No - users need link to install	Yes

For more information, please review [this blog](#) on how to publish your extensions out of the view of the public, without the need to self host your extensions.

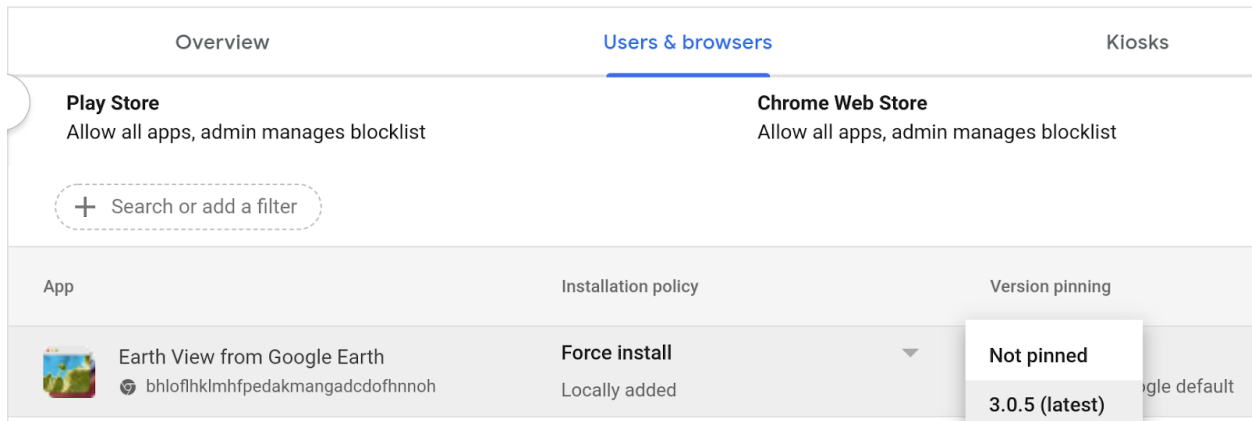
- Note that if you are managing your extensions via the admin console you need to configure the Chrome Web store permissions setting to enable private extensions to show up for your users.
 - This is found in the admin console under Devices>Chrome>Apps and Extensions>additional settings>Chrome Web Store Permissions>and set to Allow users to publish private apps that are restricted to your domain on the Chrome Web Store.

Pin an extension to a specific version in the admin console

The Google admin console now offers a few new options for extension management. First is the ability to pin to a version of an extension directly in the admin console. This provides more stability for enterprises that need to stick to a certain version of an extension. It is not recommended to pin to older versions of extensions as a best practice. If you do pin to an older version, make it a temporary measure to ensure that you have the latest functionality and security updates. This feature is only available for force-installed extensions. [For more information, please review this help center entry.](#)

1. In your Admin console, go to **Devices > Chrome> Apps and extensions>Users & browsers**
2. Select the organizational unit that contains the extension that you want to pin.
3. Select the existing extension(s) (or add a new one) you want to manage by version and under the version pinning column, select the version that you want to pin to from the drop down menu and hit save.
 - a. Note that by pinning an app or extension, it will no longer get updates, including security and compatibility updates.
 - b. You also can only pin to the current version of the extension that is present on the Chrome Web Store at the time of setup.

- c. You can also pin self-hosted apps and extensions and update the url within the admin console. See the section [Pin self-hosted apps in this help center entry](#).



The screenshot shows the 'Users & browsers' section of the Chrome Admin Console. At the top, there are three tabs: 'Overview', 'Users & browsers' (which is selected), and 'Kiosks'. Below the tabs, there are two main sections: 'Play Store' and 'Chrome Web Store', both with the policy 'Allow all apps, admin manages blocklist'. A search bar with a plus icon and the text 'Search or add a filter' is located below these sections. A table lists installed apps with columns for 'App', 'Installation policy', and 'Version pinning'. The first row shows 'Earth View from Google Earth' with a 'Force install' policy and 'Locally added' status. A dropdown menu is open for the 'Version pinning' column, showing options for 'Not pinned' and '3.0.5 (latest)'. Other options like 'Google default' are partially visible.

Version pinning in the admin console

Requirements for self hosting extensions

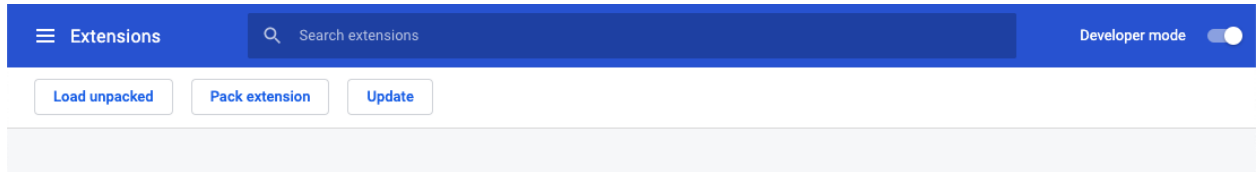
To host your extension, you will need your own web hosting services for the extension and its manifest file. This hosting location shouldn't require authentication. It needs to be accessible by devices wherever they are used. Keep this in mind if you want to host the file on your internal repository.

The steps assume that you've already created your extension, and have experience with XML files. Also that you have knowledge about group policy and using the windows registry. These steps do not apply to a 3rd party extension that you don't not develop. If you want to self-host a 3rd party extension, you should discuss this with the extension vendor directly.

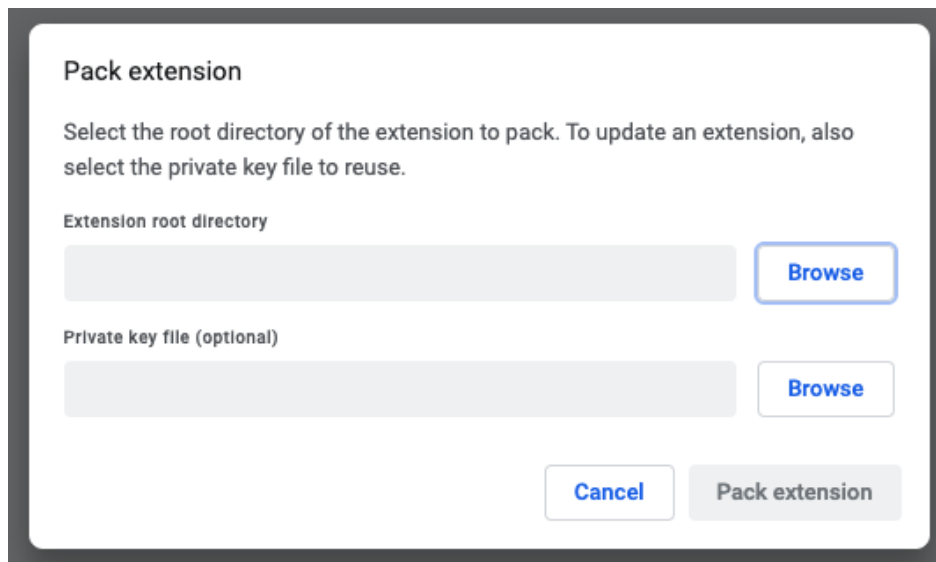
Packaging your extension

Extensions first need to be packed into a CRX file. If the extension isn't packed as a CRX file, here are the steps:

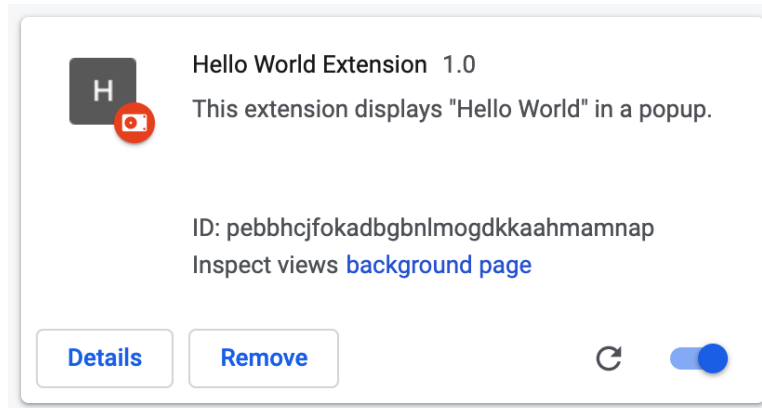
1. Go to **chrome://extensions** in the Chrome address bar and check the box for **Developer mode**.



2. After you're in developer mode, create the CRX file by clicking **Pack Extension**.



3. Select the directory where your source is. This will create your CRX file, along with a PEM file.
Top Tip: Keep the PEM file securely stored, because this is the key to your extension. You'll need it for future updates.
4. Drag the CRX into your extensions window and make sure that it loads.
 - a. Note on Windows and mac the extension will be disabled by default but on Linux it will not.
5. Test the extension and take note of the ID field and version number. These will be important later on.



5. Place the CRX file in the host location where your users or devices will download it from.
 - Note the URL of where the file is uploaded.
 - This will be important for the manifest XML file.
6. To create a manifest XML file with the app/extension ID, download URL, and version, define these 3 fields:
 - **appid** (the extension ID from step 5)
 - **codebase** (the download location for the CRX file from step 3)
 - **version** (the version of the app/extension, which should match step 5)

Example XML manifest file:

```
<?xml version='1.0' encoding='UTF-8'?>
<gupdate xmlns='http://www.google.com/update2/response' protocol='2.0'>
  <app appid='abcdefghijklmnopqrstuvwxy z123456'
  '>
    <updatecheck codebase='https://example.com/chrome/helloworld.crx'
    version='1.0' />
  </app>
</gupdate>
```

8. Upload the completed XML file to a location from where your users or devices can download it, while noting the URL.

Hosting your extension

The server that hosts your extension's .crx files must use appropriate HTTP headers to allow users to install the extension by clicking a link.

Google Chrome considers a file to be installable if either of the following is true:

- The file has the content type application/x-chrome-extension
- The file suffix is .crx and both of the following are true:
 - The file is not served with the HTTP header X-Content-Type-Options: nosniff
 - The file is served with one of the following content types:
 - empty string
 - "text/plain"
 - "application/octet-stream"
 - "unknown/unknown"
 - "application/unknown"
 - "*/*"

The most common reason for failing to recognize an installable file is that the server sends the header X-Content-Type-Options: nosniff. The second most common reason is that the server sends an unknown content type—one that isn't in the previous list. To fix an HTTP header issue, either change the configuration of the server or try hosting the .crx file at another server.

Publishing updates to your extension

Make sure that you've made the required changes to your extension and tested it. To publish updates:

1. Change the version number in your extension's manifest JSON file to a higher number.
Example:
`"version": "versionString"`
If the `"version": "1.0"`, then you can update to `"version": "1.1"` or any number higher than "1.0".
2. Update the `"version"` of `<updatecheck>` in the XML file to match the number that you put in the manifest file in the last step.
Another example:
`<updatecheck codebase='https://app.somecompany.com/chrome/helloworld.crx' version='1.1' />`
3. Recreate a CRX file which includes the new changes:
 - a. Go to **chrome://extensions** in the Chrome address bar.
 - b. Check the box for **Developer mode**.
4. Create the CRX file by clicking **Pack Extension** and selecting the directory where your source is at.
Note: For the PEM file, use the same file which was generated and saved during the first time the CRX file was created.
5. Drag the CRX into your extensions window and make sure that it loads.

6. Test the extension.
7. Replace the old CRX file and XML file with the new file.
 - a. This needs to be at the same host location from where the users or devices downloaded the files before.

The changes will be picked up during the next policy sync cycle.

Reference URLs:

- [Autoupdating](#)
- [Update URL](#)
- [Update manifest](#)

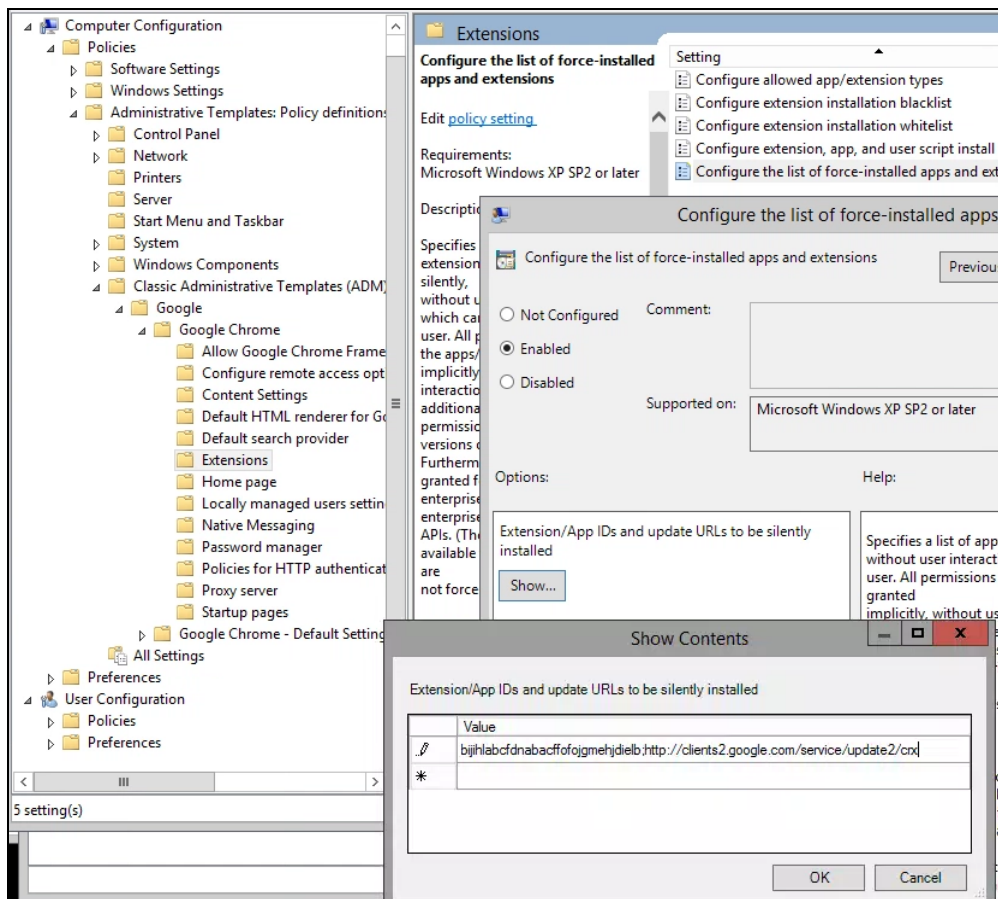
Distributing privately hosted extensions

In Group Policy: If you aren't using the Admin console, you can use the policy called "Configure the list of force installed apps and extensions" to force-install an extension on your user's device.

For privately hosted apps (not in the Chrome Web Store), use a string such as:

```
pckdojakecnhhplcgfflhndiffaohfah;https://sites.google.com/site/pushcrx/privatewebstore/extension_info.xml
```

The URL is specified to the **internal app's update.xml**, rather than the public-facing `clients2.google.com` URL.



GPO Policy “Configure the list of force-installed apps/extensions” (Show Contents)

The policies can then be applied to your chosen users, machines, or both. It can take some time for the policy to take effect. Speed things up by running "gpupdate" on your user's machine.

Manage extensions using Chrome Browser Cloud Management

Manage Chrome browser for your Windows, Mac, and Linux machines all in one place, and get an in-depth view of the state of Chrome browser in your environment. Chrome Browser Cloud Management is a great method for managing Chrome browser settings. Access to this console has no additional cost. All sections within this document that reference the Google admin console can be accessed with this feature of Chrome. With the console, you can quickly get insights on the:

- Current Chrome browser versions deployed across your fleet
- Extensions installed on each browser
- Policies being applied to each browser
- For more information about managing extensions within Chrome Browser Cloud Management, [check out this video](#)

Additional resources

Here are more resources to help you with managing the Chrome browser in your organization:

- [Chrome Browser Cloud Management landing page](#)
- [Chrome Browser Enterprise bundle](#)
- [Chrome Policy list](#)
- [Chrome Enterprise release notes](#)
- [Chrome browser update management strategies](#)
- [Chrome Enterprise Help Center](#)
- [Make Chrome default browser \(Windows 10\)](#)
- [Chrome insider blog series](#)
- [The transition of Chrome extensions to Manifest V3](#)