Guidance for the configuration of private registries for Dependabot

In this article

This article contains detailed information about configuring private registries, as well as commands you can run from the command line to configure your package managers locally.

About configuring private registries for Dependabot

This article contains recommendations and advice to help you configure Dependabot to access your private registry, along with:

- Detailed snippets of the `dependabot.yml` configuration file for each package manager.
- Important limitations or caveats.
- Steps explaining how to test that the configuration is working.
- Extra configuration options, wherever appropriate (for example, npm has a configuration file that needs to be set).
- Advice about configuring registry hosts.

You'll find detailed guidance for the setup of the following package managers and registry hosts:

- Bundler
- Docker
- Gradle
- Maven
- npm
- Nuget
- Python
- Yarn
- Artifactory
- Azure Artifacts
- GitHub Packages registry
- Nexus
- ProGet

Configuring package managers

Bundler

Supported by Artifactory, Artifacts, GitHub Packages registry, Nexus, and ProGet.

You can authenticate with either a username and password, or a token. For more information, see `ruby-gems` in "Configuration options for the dependabot.yml file."

Snippet of a `dependabot.yml` file using a username and password.

```
registries:
  ruby-example:
    type: rubygems-server
    url: https://rubygems.example.com
    username: octocat@example.com
    password: ${{secrets.MY_RUBYGEMS_PASSWORD}}
```

The snippet of `dependabot.yml` file below uses a token. For this type of registry using the GitHub Packages registry (`xyz.pkg.github.com`), the token is in fact a GitHub personal access token (PAT) .

```
registries:
  ruby-github:
    type: rubygems-server
    url: https://rubygems.pkg.github.com/octocat/github_api
    token: ${{secrets.MY_GITHUB_PERSONAL_TOKEN}}
```

Notes

Dependencies sourced directly from a GitHub repository give Dependabot access to the repository through the GitHub UI. For information about allowing Dependabot to access private GitHub dependencies, see "Allowing Dependabot to access private dependencies."

Docker

Docker supports using a username and password for registries. For more information, see `docker-registry` in "Configuration options for the dependabot.yml file."

Snippet of `dependabot.yml` file using a username and password.

```
registries:
  dockerhub:
    type: docker-registry
    url: https://registry.hub.docker.com
    username: octocat
    password: ${{secrets.MY_DOCKERHUB_PASSWORD}}
```

`docker-registry` can also be used to pull from private Amazon ECR using static AWS credentials.

```
registries:
  ecr-docker:
    type: docker-registry
    url: https://1234567890.dkr.ecr.us-east-1.amazonaws.com
    username: ${{secrets.ECR_AWS_ACCESS_KEY_ID}}
    password: ${{secrets.ECR_AWS_SECRET_ACCESS_KEY}}
```

Notes

Dependabot works with any container registries that implement the Open Container Initiative (OCI) Distribution Specification. For more information, see https://github.com/opencontainers/distribution-spec/blob/main/spec.md.

Dependabot supports authentication to private registries via a central token service or HTTP Basic Auth. For more information, see Token Authentication Specification in the Docker documentation and Basic access authentication on Wikipedia.

Limitations and workarounds

- Image names may not always be detected in Containerfiles, Helm files, or yaml files.
- Dockerfiles may only receive a version update to the first `FROM` directive.
- Dockerfiles do not receive updates to images specified with the `ARG` directive. There is a workaround available for the `COPY` directive. For more information, see https://github.com/dependabot/dependabot-core/issues/5103#issuecomment-1692420920.
- Dependabot doesn't support multi-stage Docker builds. For more information, see https://github.com/dependabot/dependabot-core/issues/7640.

Gradle

Dependabot doesn't run Gradle but supports updates to certain Gradle files. For more information, see "Gradle" in "Configuration options for the dependabot.yml file."

Gradle supports the `maven-repository` registry type. For more information, see `maven-repository` in "Configuration options for the dependabot.yml file."

The `maven-repository` type supports username and password. If the account is a GitHub account, you can use a GitHub personal access token in place of the password.

```yaml
registries:
  gradle-artifactory:
    type: maven-repository
    url: https://acme.jfrog.io/artifactory/my-gradle-registry
    username: octocat
    password: ${{secrets.MY_ARTIFACTORY_PASSWORD}}
updates:
  - package-ecosystem: "gradle"
    directory: "/"
    registries:
      - gradle-artifactory
    schedule:
      interval: "monthly"
```

Notes

You may not see all of your dependencies represented in the dependency graph, especially if some dependencies are build-time dependencies. You can use the Dependency submission API to inform GitHub about your other dependencies, and receive security updates for them. For more information, see "Using the Dependency submission API."

Maven

Maven supports username and password authentication. For more information, see `maven-repository` in "Configuration options for the dependabot.yml file."

```yaml
registries:
  maven-artifactory:
```

```
    type: maven-repository
    url: https://acme.jfrog.io/artifactory/my-maven-registry
    username: octocat
    password: ${{secrets.MY_ARTIFACTORY_PASSWORD}}
```

If the account is a GitHub account, you can use a GitHub personal access token in place of the password.

```
version: 2
registries:
  maven-github:
    type: maven-repository
    url: https://maven.pkg.github.com/octocat
    username: octocat
    password: ${{secrets.OCTOCAT_GITHUB_PAT}}
updates:
  - package-ecosystem: "maven"
    directory: "/"
    registries:
      - maven-github
    schedule:
      interval: "monthly"
```

Notes

You may not see all of your dependencies represented in the dependency graph, especially if some dependencies are build-time dependencies. You can use the Dependency submission API to inform GitHub about your other dependencies, and receive security updates for them. For more information, see "Using the Dependency submission API."

npm

You can define the configuration in the `dependabot.yml` file using the `npm-registry` type, or configure Dependabot to send all registry requests through a specified base URL.

Using the `npm-registry` type in the configuration file

You can define the private registry configuration in a `dependabot.yml` file using the `npm-registry` type. For more information, see "Configuration options for the dependabot.yml file."

The snippet of a `dependabot.yml` file below uses a token. For this type of registry using the GitHub Packages registry (`xyz.pkg.github.com`), the token is in fact a GitHub personal access token (PAT) .

```
registries:
  npm-github:
    type: npm-registry
    url: https://npm.pkg.github.com/<org-name>
    token: ${{secrets.MY_GITHUB_PERSONAL_TOKEN}}
```

The npm ecosystem requires a `.npmrc` file with the private registry URL to be checked into the repository.

Example of the content of a `.npmrc` file:

```
registry=https://<private-registry-url>/<org-name>
```

Aternatively you can add the private registry URL to an existing `.npmrc` file using the

following command.

```
npm config set registry <url>
```

For more information, see registry in the npm documentation.

You can also scope the configuration to only a single dependency or organization, in which case the token will only be valid for the organization, and different tokens can be used for different organizations for the same repository.

```
npm config set @<org-name>:registry <url>
```

This would result in a '.npmrc' with the registry:

```
@<org-name>:registry=https://<private-registry-url>/<org-name>
```

npm can be configured to use the private registry's URL in lockfiles with `replace-registry-host`. For more information, see replace-registry-host in the npm documentation.

```
npm config set replace-registry-host "never"
```

If you use `replace-registry-host`, you must locally run `npm install` in order to regenerate the lockfile to use the private registry URL. Dependabot will use the same URL when providing updates.

Once the registry is configured, you can also run `npm login` to verify that your configuration is correct and valid. The lockfile can also be regenerated to use the new private registry by running `npm install` again.

You need to ensure that the `.npmrc` file is checked into the same directory as the project's `package.json` and that the file doesn't include any environment variables or secrets. If you use a monorepo, the `.npmrc` file should live in the project's root directory.

Configuring Dependabot to send registry requests through a specified base URL

You can configure Dependabot to send all registry requests through a specified base URL. In order for Dependabot to access a public dependency, the registry must either have a cloned copy of the dependency with the requested version, or allow traffic to fetch from a public registry if the dependency is not available.

If there is no global registry defined in a `.npmrc` file, you can set `replaces-base` to `true` in the `dependabot.yml` file. For more information, see "`replaces-base`" in "Configuration options for the dependabot.yml file."

Notes

Dependencies sourced directly from a GitHub repository give Dependabot access to the repository through the GitHub UI. For information about allowing Dependabot to access private GitHub dependencies, see "Allowing Dependabot to access private dependencies."

For scoped dependencies (`@my-org/my-dep`), Dependabot requires that the private registry is defined in the project's `.npmrc file`. To define private registries for individual scopes, use `@myscope:registry=https://private_registry_url`.

Registries should be configured using the `https` protocol.

Nuget

Supported by Artifactory, Artifacts, GitHub Packages registry, Nexus, and ProGet.

The `nuget-feed` type supports username and password, or token. For more information, see `nuget-feed` in "Configuration options for the dependabot.yml file."

```
registries:
  nuget-example:
    type: nuget-feed
    url: https://nuget.example.com/v3/index.json
    username: octocat@example.com
    password: ${{secrets.MY_NUGET_PASSWORD}}
```

```
registries:
  nuget-azure-devops:
    type: nuget-feed
    url: https://pkgs.dev.azure.com/.../_packaging/My_Feed/nuget/v3/index.json
    username: octocat@example.com
    password: ${{secrets.MY_AZURE_DEVOPS_TOKEN}}
```

Notes

You can also use a token in your `dependabot.yml` file. For this type of registry using the GitHub Packages registry (`xyz.pkg.github.com`), the token is in fact a GitHub personal access token (PAT) .

```
registries:
  nuget-azure-devops:
    type: nuget-feed
    url: https://pkgs.dev.azure.com/.../_packaging/My_Feed/nuget/v3/index.json
    token: ${{secrets.MY_AZURE_DEVOPS_TOKEN}}
```

Python

Supported by Artifactory, Azure Artifacts, Nexus, and ProGet. The GitHub Packages registry is not supported.

The `python-index` type supports username and password, or token. For more information, see `python-index` in "Configuration options for the dependabot.yml file."

```
registries:
  python-example:
    type: python-index
    url: https://example.com/_packaging/my-feed/pypi/example
    username: octocat
    password: ${{secrets.MY_BASIC_AUTH_PASSWORD}}
```

```
registries:
  python-azure:
    type: python-index
    url: https://pkgs.dev.azure.com/octocat/_packaging/my-feed/pypi/example
    username: octocat@example.com
    password: ${{secrets.MY_AZURE_DEVOPS_TOKEN}}
```

```
registries:
  python-gemfury:
    type: python-index
```

```
    url: https://pypi.fury.io/my_org
    token: ${{secrets.MY_GEMFURY_TOKEN}}
```

Notes

Dependencies sourced directly from a GitHub repository give Dependabot access to the repository through the GitHub UI. For information about allowing Dependabot to access private GitHub dependencies, see "Allowing Dependabot to access private dependencies."

'url' should contain the URL, organization, and the "feed" or repository.

Yarn

The Yarn registry uses a configuration similar to that of the npm registry. For more information, see "`npm-registry`" in Configuration options for the dependabot.yml file."

```
registries:
  yarn-github:
    type: npm-registry
    url: https://npm.pkg.github.com/<org-name>
    token: ${{secrets.MY_GITHUB_PERSONAL_TOKEN}}
```

- For private registries, you have to check in a `.yarnrc.yml` file (for Yarn 3) or a `.yarnrc` file (for Yarn Classic).
- The yarn config files should not contain environment variables.
- You should configure private registries listed in the `dependabot.yml` file using `https`.

Yarn Classic

You can either specify the private registry configuration in the `dependabot.yml` file, or set up Yarn Classic according to the standard package manager instructions.

Defining the private registry configuration in the `dependabot.yml` file

You can define the private registry configuration in your `dependabot.yml` file. For more information, see "Configuration options for private registries" in "Configuration options for the dependabot.yml file."

To ensure that the private registry is listed as the dependency source in the project's `yarn.lock` file, you need to run `yarn install` on a machine with private registry access. Yarn should update the resolved field to include the private registry URL.

```
encoding@^0.1.11:
  version "0.1.13"
  resolved "https://private_registry_url/encoding/-/encoding-
0.1.13.tgz#56574afdd791f54a8e9b2785c0582a2d26210fa9"
  integrity sha512-
ETBauow1T35Y/WZMkio9jiM0Z5xjHHmJ4XmjZOq1l/dXz3lr2sRn87nJy20RupqSh1F2m3HHPSp8ShIPQJr

  dependencies:
    iconv-lite "^0.6.2"
```

Following the standard instructions from your package manager

If the `yarn.lock` file doesn't list the private registry as the dependency source, you can set up Yarn Classic according to the standard package manager instructions.

1. Define the private registry configuration in the the `dependabot.yml` file.
2. You can then either:
    - Manually set the private registry to the `.yarnrc` file by adding the registry to a `.yarnrc.yml` file in the project root with the key registry, or
    - Perform the same action by running `yarn config set registry <private registry URL>` in your terminal.

    Example of a `.yarnrc` with a private registry defined: `registry https://nexus.example.com/repository/yarn-all`

Yarn Berry (v3)

For information on the configuration, see Settings (.yarnrc.yml) in the Yarn documentation.

As with Yarn Classic, you can either specify the private registry configuration in the `dependabot.yml` file, or set up Yarn Berry according to the package manager instructions.

Defining the private registry configuration in the `dependabot.yml` file

You can define the private registry configuration in your `dependabot.yml` file. For more information, see "Configuration options for private registries" in "Configuration options for the dependabot.yml file."

To ensure the private registry is listed as the dependency source in the project's `yarn.lock` file, run `yarn install` on a machine with private registry access. Yarn should update the resolved field to include the private registry URL.

```
encoding@^0.1.11:
  version "0.1.13"
  resolved "https://private_registry_url/encoding/-/encoding-
0.1.13.tgz#56574afdd791f54a8e9b2785c0582a2d26210fa9"
  integrity sha512-
ETBauow1T35Y/WZMkio9jiM0Z5xjHHmJ4XmjZOq1l/dXz3lr2sRn87nJy20RupqSh1F2m3HHPSp8ShIPQJr

  dependencies:
    iconv-lite "^0.6.2"
```

You can also configure private registries with `npmAuthIdent` or `npmAuthToken`. For more information, see "npmAuthIdent" and "npmAuthToken" in the Yarn documentation.

```
yarn config set registry <url>
```

You can scope the configuration to only cover a single dependency or organization.

```
yarn config set @<SCOPE>:registry <url>
```

Finally, we recommend you run `yarn login` to verify that your configuration is correct and valid. The lockfile can also be regenerated to use the new private registry by running `yarn install` again.

Following the standard instructions from your package manager

If the `yarn.lock` file doesn't list the private registry as the dependency source, you can set up Yarn Berry according to the standard package manager instructions.

1. Define the private registry configuration in the the `dependabot.yml` file.
2. You can then either:
    - Manually set the private registry to the `.yarnrc` file by adding the registry to a `.yarnrc.yml` file in the project root with the key `npmRegistryServer`, or
    - Perform the same action by running `yarn config set npmRegistryServer <private registry URL>` in your terminal.

    Example of a `.yarnrc.yml` file with a private registry configured:
    `npmRegistryServer: "https://nexus.example.com/repository/yarn-all"`

    For more information, see npmRegistryServer in the Yarn documentation.

Notes

Dependencies sourced directly from a GitHub repository give Dependabot access to the repository through the GitHub UI. For information about allowing Dependabot to access private GitHub dependencies, see "Allowing Dependabot to access private dependencies."

For scoped dependencies (`@my-org/my-dep`), Dependabot requires that the private registry is defined in the project's `.yarnrc file`. To define private registries for individual scopes, use `@myscope:registry" "https://private_registry_url"`.

Configuring private registry hosts

Artifactory

For information about the configuration of Artifactory, see Configuring Artifactory in the JFrog Artifactory documentation.

Remote repositories

Remote repositories serve as a cache for build artifacts and dependencies. Instead of having to reach out to a global dependency repository, your build tool can use the artifactory cache, which will speed up build times. For more information, see Remote Repositories in the JFrog Artifactory documentation.

If you use the `replace-base` setting, you should also configure a remote repository for Artifactory if you want Dependabot to access another registry whenever the dependency isn't found in the private registry.

Virtual registry

You can use a virtual registry to group together all private and public dependencies under a single domain. For more information, see npm Registry in the JFrog Artifactory documentation.

Limitations and workarounds

The `target branch` setting does not work with Dependabot security updates on Artifactory. If you get a 401 authentication error, you need to remove the `target-branch` property from your `dependabot.yml` file. For more information, see ARTIFACTORY: Why GitHub Dependabot security updates are failing with 401 Authentication error, when it initiates a connection with Artifactory npm private registry for security updates in the JFrog Artifactory documentation.

Azure Artifacts

For information about Azure Artifacts and instructions on how to configure Dependabot to work with Azure Artifacts, see Azure DevOps in the Azure Artifacts documentation, and Use Dependabot in GitHub with Azure Artifacts, respectively.

Example of Azure Artifacts registry:

```
registries:
  nuget-azure-devops:
    type: nuget-feed
    url: https://pkgs.dev.azure.com/my_org/_packaging/public/nuget/v3/index.json
    token: ${{secrets.AZURE_DEVOPS_TOKEN}}
```

The Azure Artifacts password must be an unencoded token and should include a `:` after the token. In addition, the password cannot be base64-encoded.

You can check whether the private registry is successfully accessed by looking at the Dependabot logs.

GitHub Packages registry

For information about GitHub Packages registries, see "Working with a GitHub Packages registry." From that article, you can access pages describing how to configure the following registries.

- Bundler (rubygems)
- Docker (containers)
- GitHub Actions
- Gradle
- Maven
- npm
- NuGet
- Yarn

```
registries:
  github:
    type: npm-registry
    url: https://npm.pkg.github.com/<org-name>
    token: ${{ secrets.<token> }}
```

Notes

There is no Python container registry.

For private registries that are scoped to a particular organization, Dependabot expects the URL to include the organization name in the `dependabot.yml` file.

Nexus

For information about the configuration of Nexus, see Repository Manager 3 in the Sonatype documentation.

Notes

With Nexus Repository Pro, you can enable user tokens. For more information, see User Tokens in the Sonatype documentation.

Example of Nexus registry:

```
registries:
  npm-nexus:
    type: npm-registry
    url: https://registry.example.com/repository/npm-internal/
    token: ${{secrets.NEXUS_NPM_TOKEN}}
```

If you are running Nexus behind a reverse proxy, you need to ensure that the server is accessible using an Auth token by using `curl -v -H 'Authorization: Bearer <token>' 'https://<nexus-repo-url>/repository/<repo-name>/@<scope>%2<package>'`. For more information, see Run Behind a Reverse Proxy in the Sonatype documentation.

If you are restricting which IPs can reach your Nexus host, you need to add the Dependabot IPs to the allowlist.

- You can find the IP addresses Dependabot uses to access the registry in the meta API endpoint, under the dependabot key. For more information, see "Meta" in the Meta API documentation.
- These are the current IPs:
  - "18.213.123.130/32"
  - "3.217.79.163/32"
  - "3.217.93.44/32" For more information, see Securing Nexus Repository Manager in the Sonatype documentation.

Registries can be proxied to reach out to a public registry in case a dependency is not available in the private regstry. However, you may want Dependabot to only access the private registry and not access the public registry at all. For more information, see Quick Start Guide - Proxying Maven and NPM" in the Sonatype documentation, and "Removing Dependabot access to public registries."

ProGet

For information about ProGet and instructions on how to configure Dependabot to work with feeds in ProGet, see the ProGet documentation.

Example of ProGet registry configuration for a NuGet feed:

```
registries:
  proget-nuget-feed:
    type: nuget-feed
    url: https://proget.corp.local/nuget/MyNuGetFeed/v3/index.json
    token: ${{secrets.PROGET_APK_KEY}}
```

Example of ProGet registry configuration for Bundler (rubygems):

```
registries:
  proget-gems-feed:
    type: rubygems-server
    url: https://proget.corp.local/rubygems/MyRubygemsFeed
    token: ${{secrets.PROGET_APK_KEY}}
```

Example of ProGet registry configuration for Python (PyPI):

```
registries:
  proget-python-feed:
    type: python-index
    url: https://proget.corp.local/pypi/MyPythonFeed
    token: ${{secrets.PROGET_APK_KEY}}
```

Notes

The `token` should be an API Key with access to view packages. For more information, see API Access and API Keys in the ProGet documentation.

You can check whether the private registry is successfully accessed by looking at the Dependabot logs.