New issue

Jump to bottom

Global vs block gem server source priority doesn't work as expected #3982

Closed **deivid-rodriguez** opened this issue on Sep 30, 2020 · 2 comments · Fixed by #3655 or #4609

Labels **type: bug report**

deivid-rodriguez commented on Sep 30, 2020 Member

In a situation like the following:

```
# my-private-gem.gemspec
# ...
gem.dependency("my-another-private-gem")

# Gemfile
source 'https://my-private-server' do
  gem 'my-private-gem'
end
```

Bundler should try to resolve using the private source for my-private-gem and all of its indirect dependencies. Only if a valid resolution is not found, it should also consider `rubygems.org` for those dependencies.

Insted, Bundle will still install `my-another-private-gem` from rubygems.org if the gem exists there. We should probably search first in the same source as `my-private-gem` before falling back.

*Originally posted by **@rafaelfranca** in #3948 (comment)*

deivid-rodriguez changed the title ~~Global vs block rubygems source priority doesn't work as expected~~ **Global vs block gem sever source priority doesn't work as expected** on Sep 30, 2020

deivid-rodriguez changed the title ~~Global vs block gem sever source priority doesn't work as expected~~ **Global vs block gem server source priority doesn't work as expected** on Sep 30, 2020

deivid-rodriguez added the **type: bug report** label on Sep 30, 2020

sonalkr132 commented on Feb 11, 2021 Member

Can confirm this issue exists. bundler doc says:

> For implicit gems (dependencies of explicit gems), any source, git, or path repository declared on the parent. This results in bundler prioritizing the ActiveSupport gem from the Rails git repository over ones from rubygems.org

which is the expected behavior as per the issue but not observed.

deivid-rodriguez mentioned this issue on Feb 12, 2021

Fix source priority for transitive dependencies and split lockfile rubygems source sections #3655
Merged
4 tasks

deivid-rodriguez closed this as completed in #3655 on Feb 15, 2021

deivid-rodriguez reopened this on Feb 18, 2021

reedloden mentioned this issue on May 10, 2021

Add Bundler CVE-2020-36327 rubysec/ruby-advisory-db#468
Merged

deivid-rodriguez mentioned this issue on May 19, 2021

Fix dependency confusion issues with implicit dependencies #4609
Merged
4 tasks

deivid-rodriguez commented on May 24, 2021 Member Author

#4609 should fix this and make bundler behave as documented.

Note that the solution slightly differs from what I originally proposed in the sense that if a scoped source includes *any version* of `my-private-gem`, the gem will be picked up from there. Falling back to the default source would only happen if the name does not exist at all in the scoped source.

As I originally proposed it, the fallback would also happen if the scoped source includes `my-private-gem` but not a version that matches all requirements. With this solution, bundler will fail to resolve claiming that it couldn't find a compatible version of `my-private-gem` in `https://my-private-server`. I can't make the original proposal work since my approach is to resolve sources for each involved gem name beforehand before resolution even starts.

I think this solution matches the documentation exactly, and it's what users intuitively expect, so seems fine to me.

deivid-rodriguez closed this as completed in #4609 on May 25, 2021

Assignees

No one assigned

Labels

type: bug report

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

- **Fix source priority for transitive dependencies and split lockfile rubygems source sections**
  rubygems/rubygems
- **Fix dependency confusion issues with implicit dependencies**
  rubygems/rubygems

2 participants