59 Privilege escalation of "external user" (with maintainer privilege) to internal access through project token

Share:

TIMELINE

paxcar submitted a report to GitLab. May 12th (2 years ago)

Summary

An "external user" (a user account with the status external) which is granted "Maintainer" role on any project on the GitLab instance where "project tokens" are allowed can elevate its privilege to "Internal". An external user with maintainer permissions could create a project token, which will be connected to a bot user with internal privileges on the GitLab instance. Thus, now being able to access all internal projects and snippets as a Guest user. This includes

- Accessing all information about internal projects as if having Guest permissions (including source code)
- Creating issues on internal projects
- Creating projects and groups (these will contain no members and thus be of little use)

An external user is by the documentation described as a way to let external contractors get access to limited parts of a GitLab instance link. Stating that

Code 131 Bytes Wrap lines Copy Download

```
1 This feature may be useful when for example a contractor is working on a given project and should only have access to that project.
```

There are no warnings about giving an external user maintainer permissions. It is also possible for ANY internal user to elevate the external user to maintainer on any internal project created by that user. Thus, there is no need to ask an Admin for permission to do this. Thus, an external user (if not already granted maintainer on a project) only needs to convince one other user on the system to create a project and invite the external user as maintainer.

Steps to reproduce

1. Create a user with "external user" activated
2. Use any internal user to invite the "external user" as maintainer to a project
3. Login as the "external user" and create a project token on the project, save the token
4. Use the token to probe internal projects

Code 90 Bytes Wrap lines Copy Download

```
1  curl --header "Authorization: Bearer <TOKEN>" "https://gitlab.domain.com/api/v4/projects"
```

create groups

Code 120 Bytes Wrap lines Copy Download

```
1  curl -X POST --header "Authorization: Bearer <TOKEN>" "https://gitlab.domain.com/api/v4/groups?name=newg&path=newgroup"
```

create issues on internal projects

Code 123 Bytes Wrap lines Copy Download

```
1 curl -X POST --header "Authorization: Bearer <TOKEN>" "https://gitlab.domain.com/api/v4/projects/21/issues?title=iWasHere"
```

access source code

Code 154 Bytes Wrap lines Copy Download

```
1 curl --header "Authorization: Bearer <TOKEN>" "https://gitlab.domain.com/api/v4/projects/19/repository/blobs/83d9398518bdf1519b7b8fbbb3fa3e305a8554ef/raw"
```

Impact

An external user can access all internal projects. Thus leading to severe information disclosure and ability to interact by issues.

What is the current *bug* behavior?

An external user with maintainer privileges to a project can create a project token which is connected to a Bot with internal access.

What is the expected *correct* behavior?

The bot should not have internal access to the GitLab instance. It is stated that

Code 98 Bytes Wrap lines Copy Download

```
1 Project access tokens are scoped to a project and can be used to authenticate with the GitLab API.
```

link
Which makes it seam like the token does not have any permissions outside the project.
The bot should probably have "external privilege" as standard. At least an external user should not be able to use the bot to access internal projects.

Results of GitLab environment info

Code 1.84 KiB Wrap lines Copy Download

```
1 System information
2 System:
3 Current User:    gitlab
4 Using RVM:   no
5 Ruby Version:    3.0.1p64
6 Gem Version: /usr/lib/ruby/2.7.0/bundler/spec_set.rb:86:in `block in materialize': Could not find rake-13.0.3 in any of the sources (Bundler::GemNotFound)
7   from /usr/lib/ruby/2.7.0/bundler/spec_set.rb:80:in `map!'
8   from /usr/lib/ruby/2.7.0/bundler/spec_set.rb:80:in `materialize'
9   from /usr/lib/ruby/2.7.0/bundler/definition.rb:170:in `specs'
```

```
    from /usr/lib/ruby/2.7.0/bundler/runtime.rb:20:in `setup'
    from /usr/lib/ruby/2.7.0/bundler.rb:149:in `setup'
    from /usr/lib/ruby/2.7.0/bundler/setup.rb:20:in `block in <top (required)>'
    from /usr/lib/ruby/2.7.0/bundler/ui/shell.rb:136:in `with_level'
    from /usr/lib/ruby/2.7.0/bundler/ui/shell.rb:88:in `silence'
    from /usr/lib/ruby/2.7.0/bundler/setup.rb:20:in `<top (required)>'
    from <internal:/usr/lib/ruby/3.0.0/rubygems/core_ext/kernel_require.rb>:85:in `require'
    from <internal:/usr/lib/ruby/3.0.0/rubygems/core_ext/kernel_require.rb>:85:in `require'
Bundler Version:unknown
Rake Version:   13.0.3
Redis Version:  6.2.3
Git Version:    2.31.1
Sidekiq Version:5.2.9
Go Version: go1.16.4 linux/amd64

GitLab information
Version:    13.10.4
Revision:   e11cc45d59e
Directory:  /usr/share/webapps/gitlab
DB Adapter: PostgreSQL
DB Version: 13.2
URL:        http://gitlab.joaxcar.com
HTTP Clone URL: http://gitlab.joaxcar.com/some-group/some-project.git
SSH Clone URL:  gitlab@gitlab.joaxcar.com:some-group/some-project.git
Using LDAP: no
Using Omniauth: yes
Omniauth Providers:

GitLab Shell
Version:    13.17.0
Repository storage paths:
- default:  /var/lib/gitlab/repositories
GitLab Shell path:      /usr/share/webapps/gitlab-shell
Git:        /usr/bin/git
```

Impact

An external user can access all internal projects. Thus leading to severe information disclosure and ability to interact by issues.

The user can now

- Accessing all information about internal projects as if having Guest permissions (including source code)
- Creating issues on internal projects
- Creating projects and groups (these will contain no members and thus be of little use)