

Picture this:

- working on OSS, [crates.io](#)
- hosts Rust libraries
- found something strange
- followed to a security issue; circumstances where someone could publish to someone else's crate
- what did I do?

I kinda freaked out.

- wasn't prepared
- lots of googling
- actually more likely than I had thought – security isn't my whole job, but it is part, I do know more than the average web user
- you should be prepared too

HOW TO DISCLOSE A SECURITY VULNERABILITY

Carol (Nichols || Goulding)
@carols10cents

AGENDA

- Security issue I found
- Steps to make a disclosure
- Kinds of disclosures
- Case Studies
- To-do list

back to crates io. web app, rust backend

The screenshot shows the homepage of crates.io. At the top, there's a navigation bar with the CARGO logo, a search bar, and links for "Browse All Crates", "Docs", and "Log in with GitHub". Below the header, a large banner features the text "The Rust community's crate host" and two buttons: "Install Cargo" and "Getting Started". A paragraph below the banner explains the purpose of the site: "Instantly publish your crates and install them. Use the API to interact and find out more information about available crates. Become a contributor and enhance the site with your work." To the right of this text, there are two statistics: "96,982,277 Downloads" and "7,415 Crates in stock".

```
// TODO: this is racy
```

- improving comments like this
- changing places that did 2 queries (select + insert/update, update + insert)
- to use 9.5's UPSERT feature that does this in 1 query

```
impl User {  
    /// Updates a user or inserts a new user  
    /// into the database.  
  
    fn find_or_insert(..., username: &str, ...) {  
        ...  
        try!(conn.query("\  
            UPDATE users  
            SET gh_access_token = $1,  
                email = $2,  
                name = $3,  
                gh_avatar = $4  
            WHERE gh_username = $5  
            RETURNING *"  
            ,&[&access_token, &email, &name,  
            &avatar, &username]);  
        ...  
    }  
}
```

- gets info from github, tries to match to a user record on github username
- upsert needs a unique index
- users table didn't have a unique index on github username
- concerned with inconsistent, incorrect data
- started thinking up scenarios to get 2 user records with the same username
- end up on github help page for changing your gh username

"However, we recommend you update all existing remote repository URLs after changing your username. Because **your old username is available for use by anyone else after you change it**, that person would be able to create repositories that override the redirect entries."

<https://help.github.com/articles/what-happens-when-i-change-my-username/>

Github help for renaming your account
we had a problem, nothing to do with duplicate rows though just due to looking up user records by github username

- 1. GitHub: alice123
- 2. Publishes crate foo on crates.io
- 3. Renames GitHub account to alicerulez
- 4. Forgets about crates.io

ALICE

- will come back to forgetting in a minute

- 1. Notices Alice has renamed her GitHub account
- 2. Creates an account named alice123 & logs in to crates.io
- 3. Now has control over crate foo
- 4. Publishes a new version that eats your laundry

BOB

- 1. Uses crate foo
- 2. Notices a new version is out
- 3. Trusts Alice, upgrades and doesn't audit
- 4. No more socks :(

CAROL

let's do a little threat modeling to think about how bad this is

P(GitHub account rename) *
P(Published a crate used by others) *
P(Attacker knowing vulnerability)*
P(Attacker noticing rename)*
P(Owner not noticing)*
P(No one auditing code) = ???

- core team member had renamed their github account!
- some people had renamed their account and lost access to their crates, contacted admins who fixed the db manually
- scale of 0 to heartbleed, pretty low but not 0

so once i found this, what did i do, what should i have done?
my experience was pretty easy, going to add in tips i learned along
the way

1. STAY CALM

2. DON'T TALK ON INSECURE CHANNELS

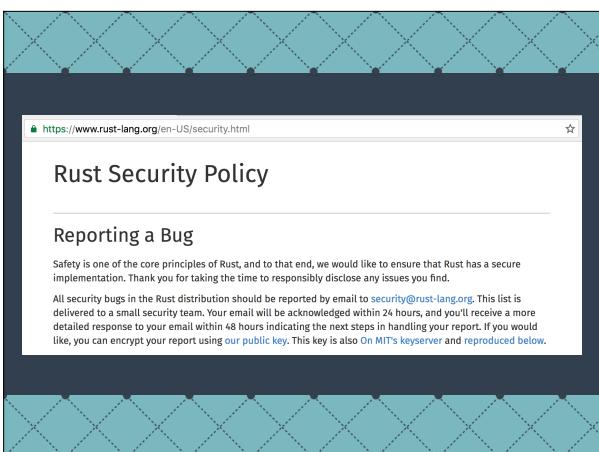
insecure: email, irc, slack, public forums, twitter
secure: signal, encrypted email



– want them to believe you and understand the severity



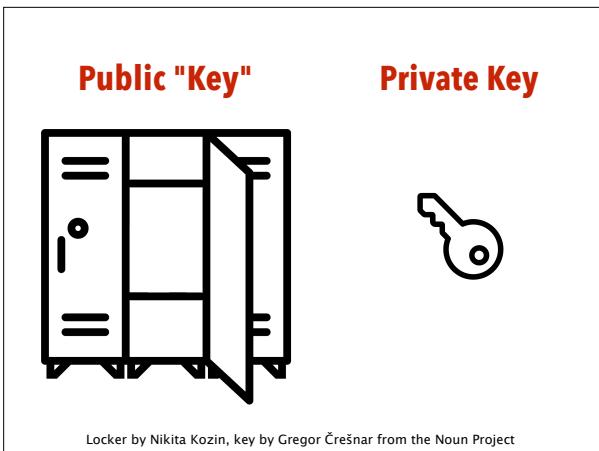
4. LOOK FOR A
SECURITY CONTACT



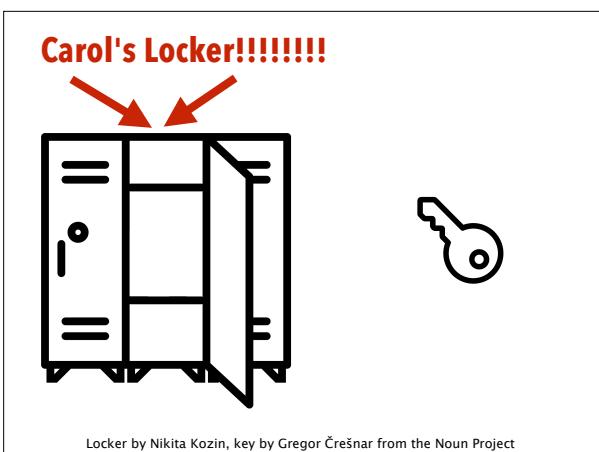
search for security, disclosure, IT dept



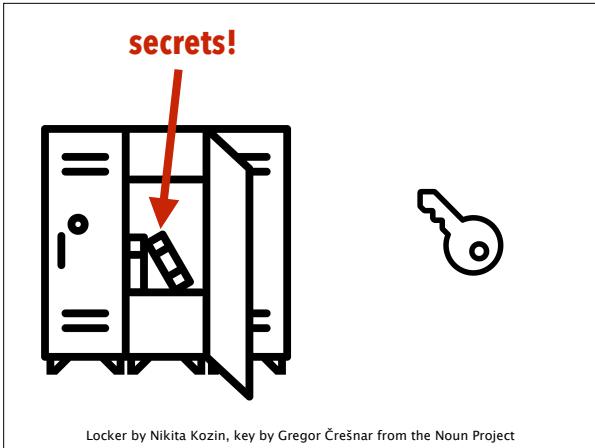
- quick, high level overview of public key encryption



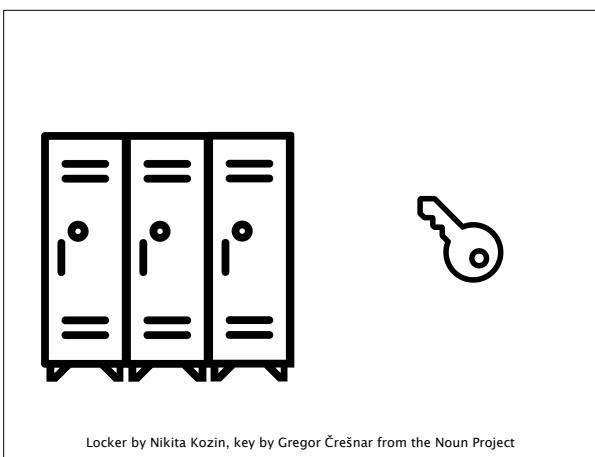
public key is more like an open locker, to which you hold the private key



You tell everyone on the internet where your locker is and that it's yours, and leave it open. You don't ever give out your key.



If someone wants to give something secret to you, they use your public key to secure the secrets, which is kind of like putting the secrets in your locker



and then shutting the door, so that only the person with the private key can get the secrets.

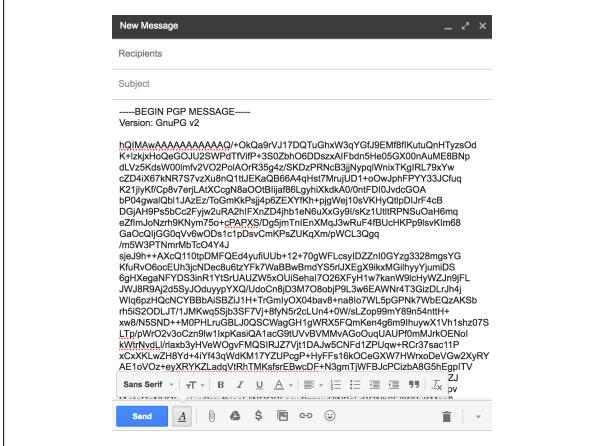
not going to go through how to generate keys, will be in the reading list

So we want to take the really good bug report and use the security team's public key to email it to them securely.

one thing I didn't realize is I didn't need to be able to encrypt ALL my email by integrating it into an email client – could just encrypt and

```
$ gpg2 --import recipients-public-key.asc
# write the body of your email in message.txt
# e = encrypt, a = ascii, r = recipient
$ gpg2 -ea -r Recipient message.txt

$ pbcopy < message.txt.asc
# paste into an email!
```



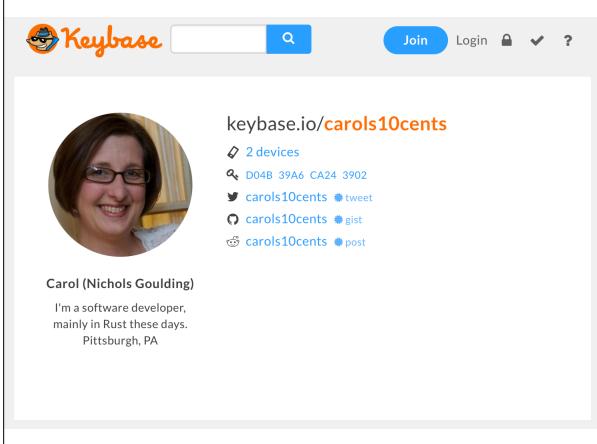
- keep in mind subject is not encrypted
- Rust team requested a meaningful subject because of spam, check guidelines
- feel like a spy



so that they can respond and ask followup questions
good idea for them to be able to verify it belongs to you



MIT key server
rust security team found it, i had forgotten but still had private key
why was i doing this on xmas 2014



The image shows a screenshot of the Keybase web interface. At the top, there's a search bar with a magnifying glass icon and a blue 'Join' button. To the right of the search bar are 'Login', a lock icon, a checkmark icon, and a question mark icon. Below the header, the URL 'keybase.io/carols10cents' is displayed. On the left, there's a circular profile picture of a woman with glasses, identified as Carol (Nichols Goulding). To the right of the profile picture, it says 'keybase.io/carols10cents' and lists '2 devices'. Below that, there are links to 'D048 39A6 CA24 3902', 'carols10cents tweet', 'carols10cents gist', and 'carols10cents post'. Underneath the profile picture, there's a bio: 'Carol (Nichols Goulding) I'm a software developer, mainly in Rust these days. Pittsburgh, PA.'

- proof that person who owns this public key controls these accounts
- not perfect but more than nothing
- once you have an encrypted reply, copy all the garbage

```
# Copy encrypted reply
# from your email into reply.txt.asc
$ pbpaste > reply.txt.asc

# d = decrypt
$ gpg2 -d reply.txt.asc

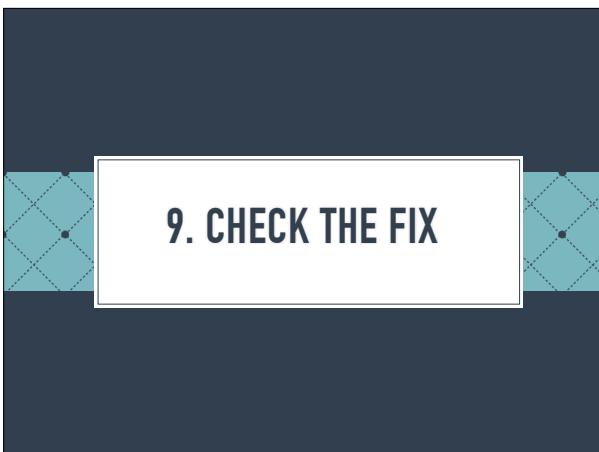
# enter your password and receive plain text!
```



- when fix will be available, when you or they will publish
- at least when they'll give you an update
- more important with less cooperative vendors



don't dream of hackers
hard to not talk about it on github, irc



- software needs to be tested
- Bug Hunters Diary – sometimes needed iterations with the vendor
- fix for my issue: add github ID to user records (doesn't change on account rename), backfill, add unique index



- You, or the vendor, or both
- inform users of risks, mitigation, any action they need to take
- share what you learned
- thought i was done learning – i was wrong

TYPES OF DISCLOSURES

 Matt McPherrin
@mcpherrinm

@rustlang @Carols10cents Please use the term "coordinated disclosure" instead of "responsible". That implies anything else is irresponsible.

6:02 PM - 17 Aug 2016

2 1 1 ...

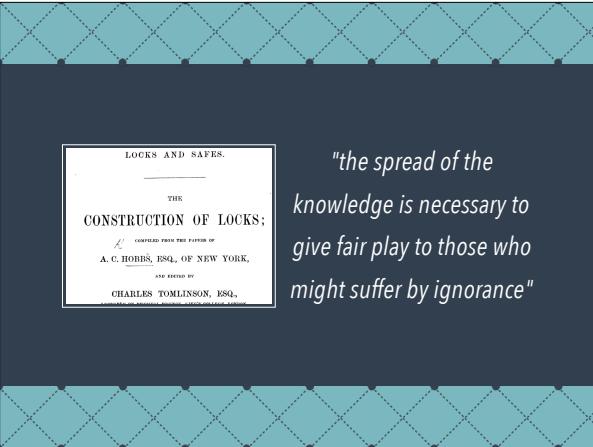
 Carol Nichols
@Carols10cents

@mcpherrinm @rustlang . . . isn't anything else irresponsible though?

 Carol Nichols
@Carols10cents

@mcpherrinm @rustlang i am half snarking and half asking for further clarification

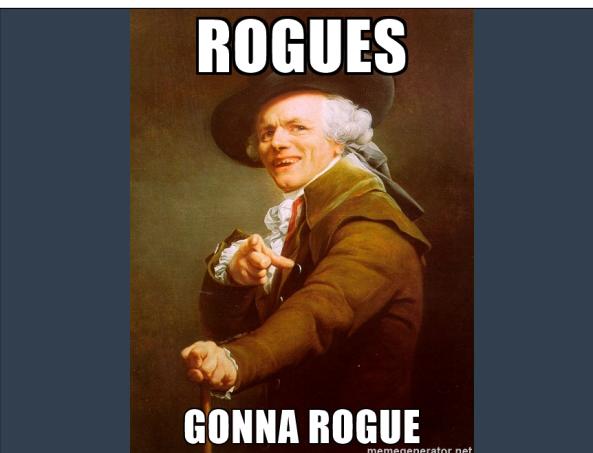
And with that, I walked into a controversy that I had no idea about: when you know about a vulnerability, what's the best thing to do with that information?



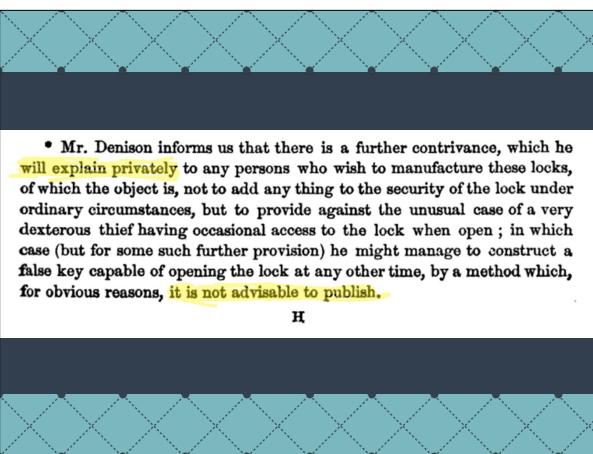
Which is a controversy that's been going on since at LEAST 1868, as documented in this book about The Construction of Locks

Great Exhibition of 1851 – picked "unpickable" locks, bug bounties, embarrassed the vendor, vendor tried to get out of paying, better locks were made

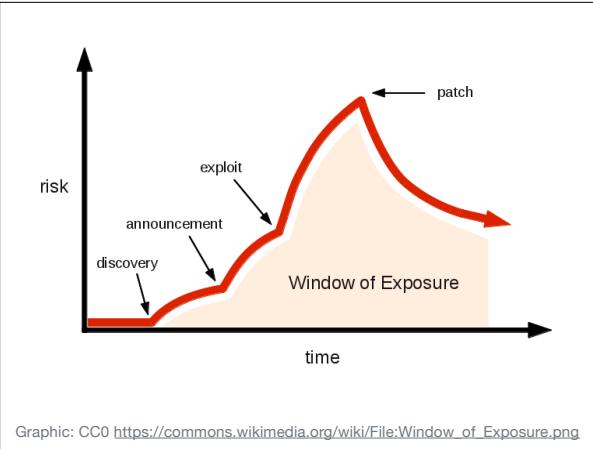
to summarize and translate to today's parlance,



this debate was not settled in the 1800s, nor was it even settled in this book



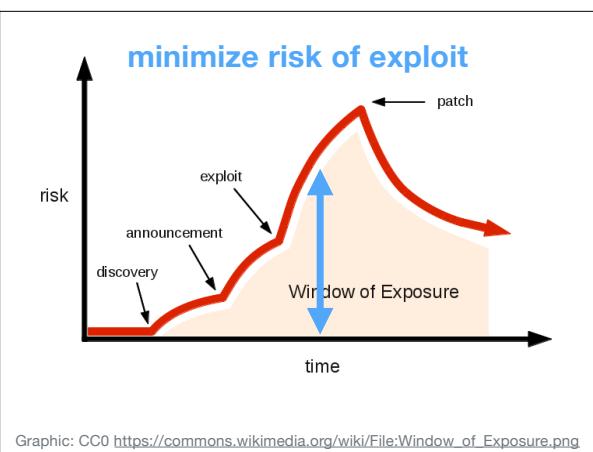
later part refused to print some details, contradicting its earlier statement



- Bruce Schneier coined term called window of exposure
- to minimize amount of time and risk users are exposed, minimize the area under the curve.
- anything trying to do that could be described as responsible
- two main ways of doing that: coordinated disclosure and full disclosure



- Scott Culp at MS coined responsible, microsoft later asked for a change to coordinated bc people argued that full disclosure could be the most responsible thing
- vendors like this most: control, save face
- bug bounty programs = carrot, we wont sue you = stick
- bug clearing houses like CERT
- sometimes vendors aren't motivated to fix

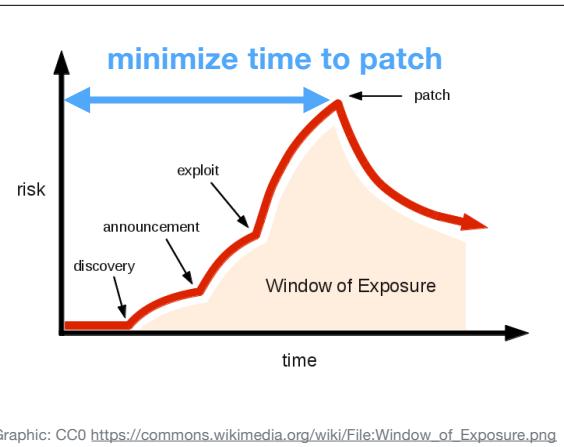


- if people don't know about it, there won't be an exploit of it
- might increase time to patch

FULL DISCLOSURE

- Tell everyone all the details immediately
- Most pressure on vendor
- Most info to users*
- Most info to exploiters**

- full disclosure: trying to minimize time to patch



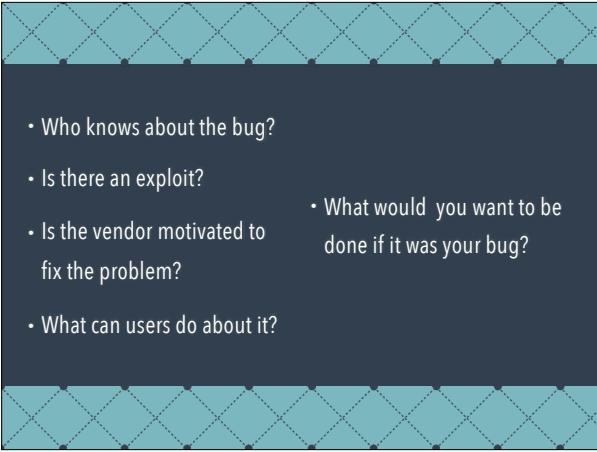
Graphic: CC0 https://commons.wikimedia.org/wiki/File:Window_of_Exposure.png

NON DISCLOSURE

- Don't publish any details, ever
- Least information to everyone

NSA likes it

deciding which kind of disclosure is an ethical decision depending on many factors

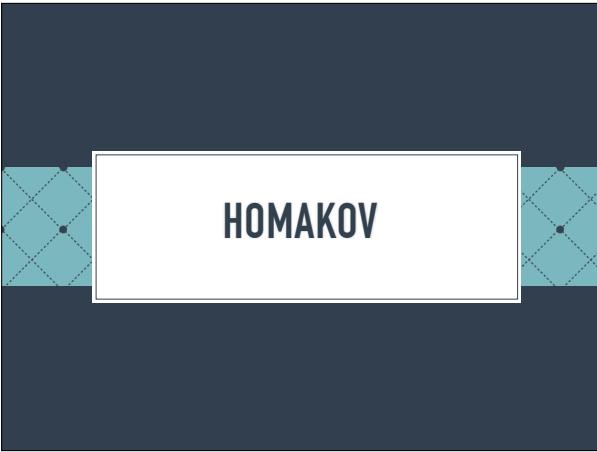


i had it easy, the vendor was people i knew and trusted

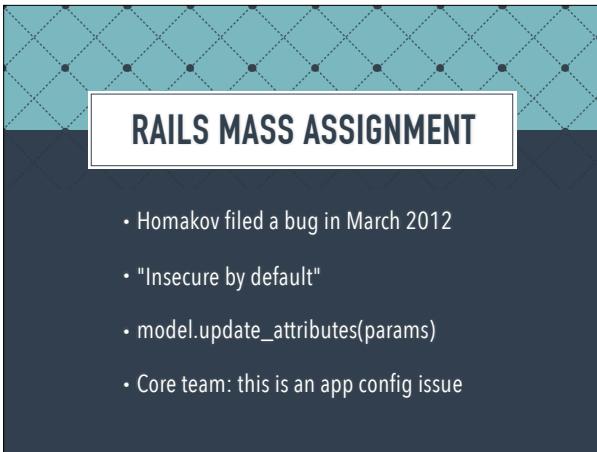
- Who knows about the bug?
- Is there an exploit?
- Is the vendor motivated to fix the problem?
- What can users do about it?
- What would you want to be done if it was your bug?



CASE STUDIES



HOMAKOV



homakov's point: no one does this app config, rails should change the default

SSH keys

New SSH key

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Title

Key

Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

Add SSH key

ⓘ Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH Problems](#).

- Homakov tested on github since they were a rails app
- tried all the forms, including the one to edit a public key in your account
- this form doesn't even exist anymore, you can only add or delete keys

```
PublicKey
:id, :user_id, :value

class PublicKeyController < ApplicationController
  def update
    @current_key = PublicKey.find_by_id params['key']['id']
    @current_key.update_attributes(params['key'])
  end
end

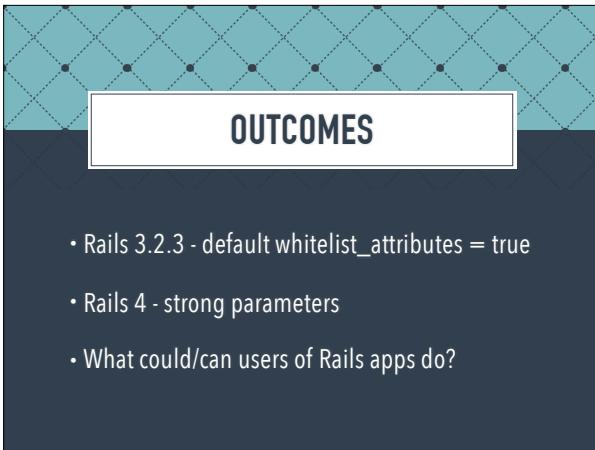
PUT github.com/public\_keys/123 // ← Homakov's key
{key: { user_id: 555 }}// ← Rails committer's user id
```

- guessed at models/params
- normally shouldn't be editing user_id, but this code happily let you
- changed a rails committer's key to homakov's

The screenshot shows a GitHub commit page for the rails/rails repository. The commit message is "wow how come I commit in master? O_o". It was made by homakov on March 4, 2012, and shows 1 changed file with 3 additions and 0 deletions. The commit content includes a diff showing changes to a file named 'hacked'.

```
+another showcase of rails apps vunlerability.  
+Github pwned. again :(  
+will you pay me for security audit?
```

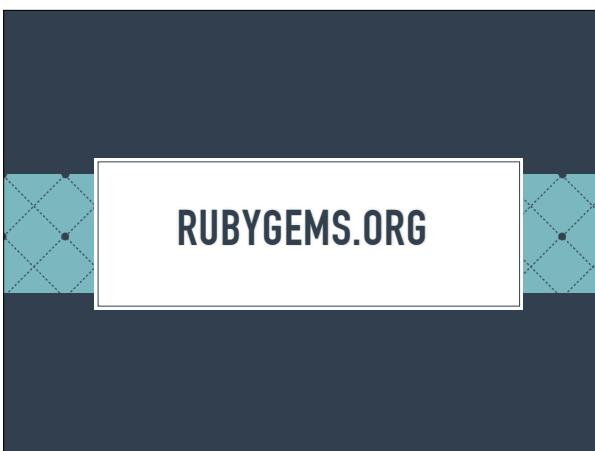
- which let him commit to master of rails. this got some attention!
- github suspended his account for a few days, gave it back
- in 2014, he updated his blog post with regrets

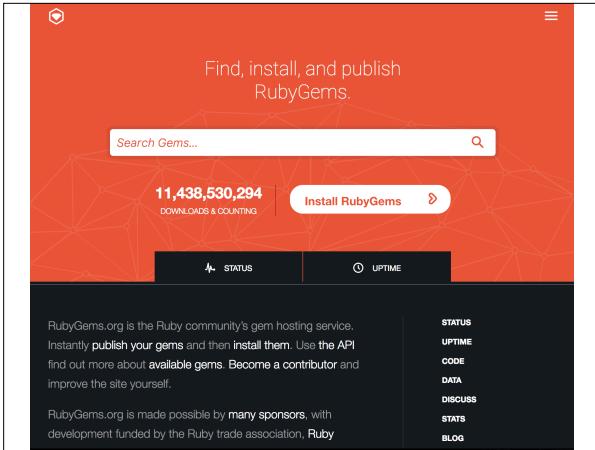


prompted the core team to address the underlying issue and change the defaults

whitelist attrs = model level
strong params = controller level

Strong parameters available as a gem later in March 2012
3.2.3: End of March 2012
Rails 4.0: June 2013
long tail of upgrading

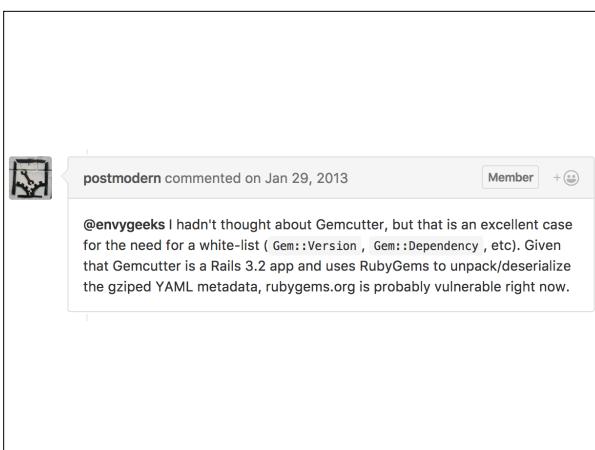




rubygems::ruby as crates.io::rust



- yaml is yet another markup language
- psych is a gem that deserializes yaml into ruby objects
- ANY ruby objects
- security announcement involving Rails parameter parsing – early Jan 2013
- by mid jan, discussion on how to make loading yaml safer on a psych issue



by jan 29, there were public discussions about the fact that
rubygems.org parses user-supplied yaml

```
--- !ruby/object:Gem::Specification
name: rails
version: !ruby/object:Gem::Version
hash: 3
prerelease:
segments:
- 3
- 1
- 0
version: 3.1.0
platform: ruby
```

- when you package a gem to be uploaded to [rubygems.org](#), the gem specification gets serialized as yaml.
- rubygems.org then needs to parse that gemspec.

blambeau commented on Jan 30, 2013 +

@postmodern @envygeeks I told the Gemcutter developers more than a week ago now. I suppose they've deployed a patch as for github's jekyll. Anyway, would you mind proposing a patch here? It's probably time to stop all of this and you seem to have sufficient knowledge for a first proposal.

postmodern commented on Jan 30, 2013 Member +

@blambeau Is there an issue open or a CVE registered? It looks like they only upgraded to Rails 3.2.11, which fixes the Rails vulnerabilities. However, rubygems.org is still reading the metadata of pushed gems.

blambeau commented on Jan 30, 2013 +

@postmodern Not that I know, but I haven't been much involved. As far as I know, they were working on a patch.

- discussions about how the [rubygems.org](#) devs had been notified
- this was at 1am EST

EXPLOIT GEM

- Contained a gemspec file with malicious YAML
- rubygems.org parsed the YAML
- Read config files with API keys, usernames, pws, and posted contents to pastie.org

- by 9:30am est, there was a gem uploaded to rubygems.org
- literally named it exploit, so it wasn't trying to be sneaky, but it was still trying to do nefarious things



Nick Quaranto
@qrush

yes, i learned this happened while on a bus ride downtown here, via Hacker News, watched years of volunteer work I've done being crucified



Especially with open source, think about the maintainers.

MITIGATION

- Shut down publishing
- All gems verified via mirrors
- Audited everything
- Rebuilt rubygems.org infrastructure

at least 2 long days of a lot of work

OUTCOMES

- rubygems.org and YAML parsing in general was made more secure, faster
- Raised awareness of the issue for other Rails apps
- What could the users of Rails apps that installed gems from Rubygems do?

NOT THE HAPPY PATH

- Sucked all around
- "Right"? "Wrong"?

- could this have been worse? could this have been better?
- what would i have done if i knew [crates.io](#) was vulnerable to a worse bug and i was having trouble getting hold of the maintainers?
- practice hypotheticals with yourself before you're panicking

TO-DO LIST

1. Make a keypair
2. Share it via keybase or a key server
3. Send an encrypted email for practice
4. Add contact info + public key to your open source projects
5. At work, add a security contact page

<https://is.gd/how2disclose>

@carols10cents

