7

HostAuthorization middleware does not suitably sanitize the Host / X-Forwarded-For header allowing redirection.

Share:

TIMELINE

tktech submitted a report to Ruby on Rails. Nov 30th (2 years ago)

When a site is configured to use the `.tkte.ch` (leading dot) short form for domain name, ex:

Code 27 Bytes Wrap lines Copy Download

```
config.hosts << '.tkte.ch'
```

it is then sanitized in sanitize_string, where it is turned into a regex:

Code 181 Bytes Wrap lines Copy Download

```
      def sanitize_string(host)
        if host.start_with?(".")
          /\A(.+\.)?#{Regexp.escape(host[1..-1])}\z/
        else
          host
        end
      end
```

The regex it is wrapped in is too permissive. It allows for things like:

Code 593 Bytes Wrap lines Copy Download

```
❯ curl -i -H "Host: google.com#sub.tkte.ch" http://localhost:3001/
HTTP/1.1 302 Found
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
Location: http://google.com#sub.tkte.ch/
Content-Type: text/html; charset=utf-8
Cache-Control: no-cache
X-Request-Id: 3b1702ac-a58f-44bf-af8a-a2933a9946fd
X-Runtime: 0.004726
Transfer-Encoding: chunked

<html><body>You are being <a href="http://google.com#sub.tkte.ch/">redirected</a>.</body></html>
```

Where the controller is simply:

Code 100 Bytes Wrap lines Copy Download

```
class RedirectController < ApplicationController
  def main
    redirect_to action: 'main'
  end
end
```

The host header poisoning was reported to us by a 3rd party researcher, and tracking it down led to this.

Impact

A user can be redirected to a hostile site.

tenderlove Ruby on Rails staff posted a comment. Dec 1st (2 years ago)

Thanks for reporting this. This definitely seems like a security issue. Do you have any suggestions for a fix? If not, I'll come up with one and post it here so we can discuss it.

tenderlove Ruby on Rails staff changed the status to • Triaged. Dec 1st (2 years ago)

tktech posted a comment. Updated Dec 2nd (2 years ago)

The fix is to validate that the `Host` you're comparing against is a valid domain name. The regex used by django is tried-and-true: `^([a-z0-9.-]+|\[[a-f0-9]*:[a-f0-9\.:]+\])(:\d+)?$`. You can find this here: https://github.com/django/django/blob/master/django/http/request.py#L37

This captures both host and port, and is well-tested. You can find the tests for valid and invalid hosts here: https://github.com/django/django/blob/master/tests/requests/tests.py#L627

tenderlove Ruby on Rails staff posted a comment. Jan 5th (2 years ago)

@tktech that seems fine to me. Can you make a patch against Rails that uses that regex? Thank you!

tktech invited another hacker as a collaborator. Jan 6th (2 years ago)

Code 1.96 KiB

Wrap lines Copy Download

```
diff --git a/actionpack/lib/action_dispatch/middleware/host_authorization.rb b/actionpack/lib/action_dispatch/middleware/host_authorization.rb
index 4564bdafe0..291fdee864 100644
--- a/actionpack/lib/action_dispatch/middleware/host_authorization.rb
+++ b/actionpack/lib/action_dispatch/middleware/host_authorization.rb
@@ -103,11 +103,20 @@ def call(env)

     private
       def authorized?(request)
-        origin_host = request.get_header("HTTP_HOST").to_s.sub(/:\d+\z/, "")
-        forwarded_host = request.x_forwarded_host.to_s.split(/,\s?/).last.to_s.sub(/:\d+\z/, "")
-
-        @permissions.allows?(origin_host) &&
-          (forwarded_host.blank? || @permissions.allows?(forwarded_host))
+        valid_host = /
+          \A
+          (?<host>[a-z0-9.-]+|\[[a-f0-9]*:[a-f0-9\.:]+\])
+          (:\d+)?
+          \z
+        /x
+
+        origin_host = valid_host.match(
+          request.get_header("HTTP_HOST").to_s.downcase)
+        forwarded_host = valid_host.match(
+          request.x_forwarded_host.to_s.split(/,\s?/).last)
+
+        origin_host && @permissions.allows?(origin_host[:host]) && (
+          forwarded_host.nil? || @permissions.allows?(forwarded_host[:host]))
       end

       def excluded?(request)
diff --git a/actionpack/test/dispatch/host_authorization_test.rb b/actionpack/test/dispatch/host_authorization_test.rb
index 79240ab9b1..0ebb09b26b 100644
--- a/actionpack/test/dispatch/host_authorization_test.rb
+++ b/actionpack/test/dispatch/host_authorization_test.rb
@@ -226,4 +226,15 @@ class HostAuthorizationTest < ActionDispatch::IntegrationTest
       ActionDispatch::HostAuthorization.new(App, "example.com", ->(env) { true })
     end
   end
+
+  test "only compare to valid hostnames" do
+    @app = ActionDispatch::HostAuthorization.new(App, ".example.com")
+
+    get "/", env: {
+      "HOST" => "example.com#sub.example.com",
+    }
+
+    assert_response :forbidden
+    assert_match "Blocked host: example.com#sub.example.com", response.body
+  end
 end
```

tenderlove Ruby on Rails staff posted a comment. Feb 9th (2 years ago)

@tktech thanks for the patch! I'll apply it and make sure the tests pass then cut a security release.

tenderlove Ruby on Rails staff updated CVE reference to CVE-2021-22881. Feb 10th (2 years ago)

rafaelfranca Ruby on Rails staff closed the report and changed the status to Resolved. Feb 10th (2 years ago)

The issue was fixed and released in Rails 6.0.3.5 and 6.1.2.1. Thank you for reporting and working with us in the fix.

The Internet Bug Bounty rewarded davenorth with a $5 bounty. Feb 10th (2 years ago)

The Internet Bug Bounty rewarded tktech with a $495 bounty. Feb 10th (2 years ago)

rafaelfranca Ruby on Rails staff requested to disclose this report. Feb 10th (2 years ago)

tktech agreed to disclose this report. Feb 10th (2 years ago)

This report has been disclosed. Feb 10th (2 years ago)