Ruby master - Feature #12558

Ruby does not support alternative server name in SSL connection

07/06/2016 08:12 AM - gangshen (gang shen)

Status:	Rejected
Priority:	Normal
Assignee:	naruse (Yui NARUSE)
Target version:	

Description

In server keystore file, it specifies Owner like Owner: CN=default.server. When used SSL connection to connect to the server, it reported a error #. I checked the ruby code and found that it seemed Ruby did not support this scenario when an Owner is specified in keystore file and its value is not the server real hostname or IP address.

First, lets have a look at the error message and call stacks

```
(Time: 2016-07-06 09:33:51 +0800)(PID: 30558)(LVL: ERROR) -: Failed to POST data due to error: #<O
penSSL::SSL::SSLError: hostname "xxx.xx.xxx.xxx" does not match the server certificate>
(Time: 2016-07-06 09:33:51 +0800)(PID: 30558)(LVL: ERROR) -: Trace: ["/usr/local/ruby/lib/ruby/sit
e_ruby/2.1.0/openssl/ssl.rb:139:in `post_connection_check'", "/usr/local/ruby/lib/ruby/2.1.0/net/h
ttp.rb:922:in `connect'", "/usr/local/ruby/lib/ruby/2.1.0/net/http.rb:863
```

From code /usr/local/ruby/lib/ruby/2.1.0/net/http.rb:922
it calls s.post_connection_check(@address) function which raise the error I mentioned. This function takes @address as the parameters which is passed by initialize function as below. From the description of the initialize function, the address should be a DNS hostname or IP address.

```
# Creates a new Net::HTTP object for the specified server address,
# without opening the TCP connection or initializing the HTTP session.
# The +address+ should be a DNS hostname or IP address.
def initialize(address, port = nil)
  @address = address
```

/usr/local/ruby/lib/ruby/2.1.0/net/http.rb:922

```
  if use_ssl?
    begin
      if proxy?
        buf = "CONNECT #{@address}:#{@port} HTTP/#{HTTPVersion}\r\n"
        buf << "Host: #{@address}:#{@port}\r\n"
        if proxy_user
          credential = ["#{proxy_user}:#{proxy_pass}"].pack('m')
          credential.delete!("\r\n")
          buf << "Proxy-Authorization: Basic #{credential}\r\n"
        end
        buf << "\r\n"
        @socket.write(buf)
        HTTPResponse.read_new(@socket).value
      end
      s.session = @ssl_session if @ssl_session
      # Server Name Indication (SNI) RFC 3546
      s.hostname = @address if s.respond_to? :hostname=
      Timeout.timeout(@open_timeout, Net::OpenTimeout) { s.connect }
      if @ssl_context.verify_mode != OpenSSL::SSL::VERIFY_NONE
        s.post_connection_check(@address)
      end
      @ssl_session = s.session
    rescue => exception
      D "Conn close because of connect error #{exception}"
      @socket.close if @socket and not @socket.closed?
      raise exception
    end
  end
  on_connect
```

```
end
```

Then take a look at s.post_connection_check(@address) function in /usr/local/ruby/lib/ruby/site_ruby/2.1.0/openssl/ssl.rb,

```
  def post_connection_check(hostname)
    unless OpenSSL::SSL.verify_certificate_identity(peer_cert, hostname)
      raise SSLError, "hostname \"#{hostname}\" does not match the server certificate"
    end
    return true
  end
```

it calls verify_certificate_identity(peer_cert, hostname) as below:

```
def verify_certificate_identity(cert, hostname)
  should_verify_common_name = true
  cert.extensions.each{|ext|
    next if ext.oid != "subjectAltName"
    ostr = OpenSSL::ASN1.decode(ext.to_der).value.last
    sequence = OpenSSL::ASN1.decode(ostr.value)
    sequence.value.each{|san|
      case san.tag
      when 2 # dNSName in GeneralName (RFC5280)
        should_verify_common_name = false
        reg = Regexp.escape(san.value).gsub(/\\\*/, "[^.]+")
        return true if /\A#{reg}\z/i =~ hostname
      when 7 # iPAddress in GeneralName (RFC5280)
        should_verify_common_name = false
        # follows GENERAL_NAME_print() in x509v3/v3_alt.c
        if san.value.size == 4
          return true if san.value.unpack('C*').join('.') == hostname
        elsif san.value.size == 16
          return true if san.value.unpack('n*').map { |e| sprintf("%X", e) }.join(':') == hostname
        end
      end
    }
  }
  if should_verify_common_name
    cert.subject.to_a.each{|oid, value|
      if oid == "CN"
        reg = Regexp.escape(value).gsub(/\\\*/, "[^.]+")
        return true if /\A#{reg}\z/i =~ hostname
      end
    }
  end
  return false
end
```

As you can see from above code, there is options that can set a alternative server hostname so that SSL verification can pass.

History

#1 - 07/06/2016 03:16 PM - rhenium (Kazuki Yamaguchi)

- Description updated

FWIW, it looks like there was Net::HTTP#enable_post_connection_check option (r13499, 2007-09-23), but it was removed soon afterward (r13648, 2007-10-07).

#2 - 06/26/2019 04:56 AM - jeremyevans0 (Jeremy Evans)

- Backport deleted (2.1: UNKNOWN, 2.2: UNKNOWN, 2.3: UNKNOWN)

- Assignee set to naruse (Yui NARUSE)

- Status changed from Open to Assigned

- Tracker changed from Bug to Feature

- File net-http-ssl-verification-hostname.patch added

This isn't a bug, this is a request for a feature that doesn't currently exist, which is the ability to connect to a server with an SSL certificate that doesn't match the address you are using to connect to the server. While this is in general something you should only do in rare cases, I am sure there are cases where it is useful. Attached is a patch that implements this feature, using a Net::HTTP#ssl_verification_hostname accessor. Example:

```
http = Net::HTTP.new('172.217.6.78', 443)
http.use_ssl = true
http.get '/'
# OpenSSL::SSL::SSLError

http.ssl_verification_hostname = 'www.google.com'
http.get '/'
# => #<Net::HTTPMovedPermanently 301 Moved Permanently readbody=true>
```

#3 - 12/12/2019 05:04 PM - jeremyevans0 (Jeremy Evans)

- Status changed from Assigned to Rejected

I believe this feature is not needed anymore. Feature #15215 was implemented which supports setting an ipaddr separately from the host name, so you can connect to any IP address and still use the host name passed to Net::HTTP.new.

Files

net-http-ssl-verification-hostname.patch	2.28 KB	06/26/2019	jeremyevans0 (Jeremy Evans)