

# [UNIX] Perl Format String Integer Wrap

---

*Source:* <http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-12/msg00103.html>

---

- *From:* SecuriTeam <[support@xxxxxxxxxxxxxx](mailto:support@xxxxxxxxxxxxxx)>
  - *Date:* 28 Dec 2005 12:14:18 +0200
- 

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

-----

Perl Format String Integer Wrap

---

## SUMMARY

Perl suffers from an integer wrap overflow inside the explicit parameter format string functionality, this has been confirmed to be a vector for remote code execution.

## DETAILS

Vulnerable Systems:

- \* Perl version 5.9.2
- \* Perl 5.8.6

Value over INT\_MAX(value of I) inside explicit parameter format string (%I\$n) causes integer wrap in the efix (32bit signed integer) variable inside the function Perl\_sv\_vcatpvfn (see example 1) (sv.c:~9360). Allowing for a write value anywhere in memory exploitation vector (see example 2). Further, heap corruption itself is possible (see example 3), as are more exotic non-reliable \$PC redirection (see example 4). From what we have seen the first exploitation method is the only valid one. Perl itself is not directly vulnerable to remote attacks due to this flaw, however any perl program with format string vulnerabilities is. The vulnerability is not to limited DoS (as reported previously) but remote

## [UNIX] Perl Format String Integer Wrap

code execution as well as information leakage and DoS.

Perl itself is not generally impacted by this vulnerability, but programs with format string vulnerabilities (Dyad Security has confirmed that several programs available at this time have this specific issue) can be vulnerable to remote code execution. Information about creating a robust generic exploit is forthcoming, so public knowledge of exploitation methods for this issue is in the cards.

Perl 5.9.2 and perl 5.8.6 have been tested and found to be vulnerable on linux, freebsd, dragonflybsd on the ia32 platform. It is assumed that a much larger range of software and platforms are also affected, as the sv.c seems to remain seemingly static over time, however this is not confirmed.

Example 1:

```
$ gdb myperl/bin/perl5.8.7
```

```
GNU gdb 6.3
```

```
Copyright 2004 Free Software Foundation, Inc.
```

```
GDB is free software, covered by the GNU General Public License, and you are
```

```
welcome to change it and/or distribute copies of it under certain conditions.
```

```
Type "show copying" to see the conditions.
```

```
There is absolutely no warranty for GDB. Type "show warranty" for details.
```

```
This GDB was configured as "i686-pc-linux-gnu"...Using host libthread_db library "/lib/tls/libthread_db.so.1".
```

```
(gdb) break sv.c:9232
```

```
Breakpoint 1 at 0x80c0df0: file sv.c, line 9232.
```

```
(gdb) set args -e 'printf("%2147483647\n");'
```

```
(gdb) run
```

```
Breakpoint 1, Perl_sv_vcatpvfn (sv=0x812d180, pat=0x0, patlen=0, args=0x0, svargs=0x8133080,
```

```
svmax=0, maybe_tainted=0xbffb72cb "") at sv.c:9232
```

```
9232 in sv.c
```

```
(gdb) p efix
```

```
$1 = 2147483647
```

```
(gdb) set args -e 'printf("%2147483648\n");'
```

```
(gdb) run
```

```
Breakpoint 1, Perl_sv_vcatpvfn (sv=0x812d180,
```

```
pat=0x80000000 <Address 0x80000000 out of bounds>, patlen=0, args=0x0, svargs=0x8133080,
```

```
svmax=0, maybe_tainted=0xbfb0640b "") at sv.c:9232
```

```
9232 in sv.c
```

```
(gdb) p efix
```

```
$2 = -2147483648
```

```
(gdb) cont
```

## [UNIX] Perl Format String Integer Wrap

Modification of a read-only value attempted at -e line 1.

Program exited with code 0377.

```
(gdb) set args -e 'printf("%2147483649$n");'
```

```
(gdb) run
```

```
Breakpoint 1, Perl_sv_vcatpvfn (sv=0x812d180,  
pat=0x80000001 <Address 0x80000001 out of bounds>, patlen=0, args=0x0,  
svargs=0x8133080,  
svmax=0, maybe_tainted=0xbfe69b9b ""') at sv.c:9232  
9232 in sv.c
```

```
(gdb) p efix
```

```
$3 = -2147483647
```

```
(gdb) cont
```

Program received signal SIGSEGV, Segmentation fault.

```
Perl_sv_setiv (sv=0x0, i=0) at sv.c:1652
```

```
1652 in sv.c
```

```
(gdb) bt
```

```
#0 Perl_sv_setiv (sv=0x0, i=0) at sv.c:1652
```

```
#1 0x080b6349 in Perl_sv_setuv_mg (sv=0x0, u=0) at sv.c:1743
```

```
#2 0x080c0e06 in Perl_sv_vcatpvfn (sv=0x812d180,  
pat=0x80000001 <Address 0x80000001 out of bounds>, patlen=0, args=0x0,  
svargs=0x8133080,  
svmax=0, maybe_tainted=0xbfe69b9b ""') at sv.c:9232
```

```
#3 0x080e923b in Perl_do_sprintf (sv=0x812d180, len=1, sarg=0x813307c) at  
doop.c:713
```

```
#4 0x080de48a in Perl_pp_prtf () at pp_sys.c:1489
```

```
#5 0x080ad038 in Perl_runops_standard () at run.c:37
```

```
#6 0x080615c7 in S_run_body (oldscope=1) at perl.c:2000
```

```
#7 0x080613ff in perl_run (my_perl=0x812d008) at perl.c:1919
```

```
#8 0x0805e61f in main (argc=3, argv=0xbfe69da4, env=0xbfe69db4) at  
perlmain.c:98
```

```
(gdb) x/i $eip
```

```
0x80b61a8 <Perl_sv_setiv+8>: mov 0x8(%ebx),%edx
```

```
(gdb) i r ebx edx
```

```
ebx 0x0 0
```

```
edx 0x812d180 135451008
```

```
(gdb)
```

Example 2:

```
#0 Perl_sv_setiv (sv=0x815f821, i=0) at sv.c:2184
```

```
2184 SvIVX(sv) = i;
```

```
(gdb) x/i $eip
```

```
0x80c815c <Perl_sv_setiv+108>: mov %esi,0xc(%eax)
```

Example 3:

```
#0 0xb7e69fb0 in malloc_consolidate () from /lib/tls/libc.so.6
```

EXAMPLE 4:

```
#0 0x09010e50 in ?? ()
```

[UNIX] Perl Format String Integer Wrap

Patch:

Unofficial patch exists on:

<[http://www.dyadsecurity.com/advisory/perl/perl-5.9.2-exp\\_parameter\\_intwrap\\_vulnerability.patch](http://www.dyadsecurity.com/advisory/perl/perl-5.9.2-exp_parameter_intwrap_vulnerability.patch)>  
[http://www.dyadsecurity.com/advisory/perl/perl-5.9.2-exp\\_parameter\\_intwrap\\_vulnerability.patch](http://www.dyadsecurity.com/advisory/perl/perl-5.9.2-exp_parameter_intwrap_vulnerability.patch)

ADDITIONAL INFORMATION

The information has been provided by <<mailto:robert@xxxxxxxxxxxxxxxxxxxx>>  
Robert.

The original article can be found at:  
<<http://www.dyadsecurity.com/perl-0002.html>>  
<http://www.dyadsecurity.com/perl-0002.html>

=====

This bulletin is sent to members of the SecuriTeam mailing list.  
To unsubscribe from the list, send mail with an empty subject line and body to:  
list-unsubscribe@xxxxxxxxxxxxxxxxxx  
In order to subscribe to the mailing list, simply forward this email to: list-subscribe@xxxxxxxxxxxxxxxxxx

=====  
=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.  
In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.



- Prev by Date: *[\[EXPL\] Windows Metafile mtNoObjects \(MS05-053, DoS, Exploit\)](#)*
- Next by Date: *[\[NT\] mIRC Local Buffer Overflow \(DDC Filter\)](#)*
- Previous by thread: *[\[EXPL\] Windows Metafile mtNoObjects \(MS05-053, DoS, Exploit\)](#)*
- Next by thread: *[\[NT\] mIRC Local Buffer Overflow \(DDC Filter\)](#)*
- Index(es):
  - ◆ *[Date](#)*
  - ◆ *[Thread](#)*