bugs found by our scanner #1236

Closed lqiulin opened this issue on Feb 20, 2019 · 3 comments

Labels bug in development

lqiulin commented on Feb 20, 2019

Hi, we developed a taint analysis based static analysis tool named Vanguard. It could prognosis potential vulnerabilities by identifying security-sensitive operations (e.g. divide-zero, mod-zero, array-index-access, and sensitive function calls) without proper checks for their operands.

Some code locations are listed in the following. We think these locations maybe bugs after our manual analysis. Please check them, and add precondition checks if necessary.

Divide/Mod-Zero

1.in function `vips_zoom_gen` , zoom.c#L260#L266#L275

```
left = VIPS_ROUND_DOWN( r->left, zoom->xfac );
right = VIPS_ROUND_UP( ri, zoom->xfac );
top = VIPS_ROUND_DOWN( r->top, zoom->yfac );
bottom = VIPS_ROUND_UP( bo, zoom->yfac );
```

```
s.left = left / zoom->xfac;
s.top = top / zoom->yfac;
s.width = width / zoom->xfac;
s.height = height / zoom->yfac;
```

```
left = VIPS_ROUND_UP( r->left, zoom->xfac );
right = VIPS_ROUND_DOWN( ri, zoom->xfac );
top = VIPS_ROUND_UP( r->top, zoom->yfac );
bottom = VIPS_ROUND_DOWN( bo, zoom->yfac );
```

Divisor: zoom->xfac, zoom->yfac
Result: Could be 0, Please Check.

2.in function `vips_point_build` , point.c#L105

```
float range = max - min;
if( vips_linear1( in, &t[2],
        255.0 / range, -min * 255.0 / range,
        "uchar", TRUE,
        NULL ) )
        return( -1 );
in = t[2];
```

Divisor: range
Result: Could be 0, Please Check.

3.in function `vips_eye_point` , eye.c#L83

```
double h = ((point->height - 1) * (point->height - 1));
return( y * y * cos( c * x * x ) / h );
```

Divisor: h
Result: Could be 0, Please Check.

4.in function `vips_mask_point` , mask.c#L85

```
dx = (double) x / half_width;
dy = (double) y / half_height;
```

Divisor: half_width, half_height
Result: Could be 0, Please Check.

Array-Index-Bound

1.in function `vips_gamma_build` , gamma.c#L97

```
scale = pow( vips_gamma_maxval[in->BandFmt],
                1.0 / gamma->exponent ) /
                vips_gamma_maxval[in->BandFmt];
```

Array expression: vips_gamma_maxval[in->BandFmt]
needs bound checking: 0<=in->BandFmt<10

2.in function `vips_byteswap_gen` , byteswap.c#L138

```
SwapFn swap = vips_byteswap_swap_fn[im->BandFmt];
```

Array expression: vvips_byteswap_swap_fn[im->BandFmt]
needs bound checking: 0<=in->BandFmt<10

3.in function `vips_byteswap_build` ,byteswap.c#L169

```
if( byteswap->in->Coding != VIPS_CODING_NONE ||
                !vips_byteswap_swap_fn[byteswap->in->BandFmt] )
                return( vips_image_write( byteswap->in, conversion->out ) );
```

Array expression: vips_byteswap_swap_fn[im->BandFmt]
needs bound checking: 0<=in->BandFmt<10

Sensitive-Function-Call

1.in function `find_header` , unpack_seek.c#L289

```
memcpy (wphdr, sp - 4, sizeof (*wphdr));
```

[memcpy] is a security-sensitive function using tainted data: [wphdr]

2.in function `rtiff_memcpy_line` , tiff2vips.c#L1219

```
memcpy( q, p, len );
```

[memcpy] is a security-sensitive function using tainted data: [len]

3.in function `tile_copy` , sinkscreen.c#L843

```
memcpy( q, p, len );
```

[memcpy] is a security-sensitive function using tainted data: [len]

4.in function `vips_region_paint` , region.c#L958#L987

```
memset( (char *) q, value, wd );

memcpy( (char *) q1, (char *) q, wd );
```

[memset] is a security-sensitive function using tainted data: [wd]
[memcpy] is a security-sensitive function using tainted data: [wd]

jcupitt added a commit that referenced this issue on Feb 20, 2019

`prevent /0 in eye for width/height 1` ··· `2fb81b8`

jcupitt added a commit that referenced this issue on Feb 20, 2019

`prevent /0 in freq mask for very small masks` ··· `65a259a`

jcupitt commented on Feb 20, 2019 Member

Hello **@lqiulin**, thank you very much for testing libvips.

I've looked through the problems you found:

Divide by zero

1. in `zoom.c` , `xfac` / `yac` are guaranteed non-zero by line 383 etc. The last three arguments to `VIPS_ARG_INT` give the minimum, maximum and default values.
2. `max` and `min` here are class variables. They are only ever set to constant values in class init, so can't be equal.
3. You're right, there's a possible /0 here. Fixed in git master, thanks!
4. Again, fixed in git master, thanks!

Array index

1. `BandFmt` is guaranteed to be in the correct range by the use of `ARG_ENUM` at line 1131 in `image.c` , so I think all these are OK.

Sensitive function call

1. This is for another project, I think.
2. `sfn` is called at lines 1591 and 1939, and both calls have checked values for len, so I think this is safe.
3. I guess these are triggered by `VIPS_IMAGE_SIZEOF_PEL` doing a lookup on `BandFmt` . Again, the `ARG_ENUM` makes this safe.

Thanks again!

jcupitt added bug **in development** labels on Feb 20, 2019

jcupitt commented on Mar 1, 2019 Member

I think this is done, I'll close. Thank you again for reporting this.

jcupitt closed this as completed on Mar 1, 2019

lovell commented on Jul 16, 2021 Member

It looks like this has been assigned CVE-2021-27847 however the version number is incorrect in the report.

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-27847

For anyone visiting this issue as a result of the above, the problem was fixed in v8.8.0 and is only present in v8.7.4 (Jan 2019) and earlier.

Assignees

No one assigned

Labels

bug in development

Projects

None yet

Milestone

No milestone

Development

No branches or pull requests

3 participants