CVE-2014-9825

描述：Heap-based buffer overflow in ImageMagick allows remote attackers to have unspecified impact via a crafted psd file, a different vulnerability than CVE-2014-9824.

软件：Imagemagick

源码：coders/psd.c

出现位置：

关键源码：

diff --git a/coders/psd.c b/coders/psd.c

index 985e32d..d818fe3 100644

--- a/coders/psd.c

+++ b/coders/psd.c

@@ -799,7 +799,7 @@ static MagickStatusType ReadPSDChannelRaw(Image \*image,const size\_t channels,

" layer data is RAW");

row\_size=GetPSDRowSize(image);

- pixels=(unsigned char \*) AcquireQuantumMemory(row\_size,sizeof(\*pixels));

+ pixels=(unsigned char \*) AcquireQuantumMemory(8\*row\_size,sizeof(\*pixels));

if (pixels == (unsigned char \*) NULL)

ThrowBinaryException(ResourceLimitError,"MemoryAllocationFailed",

image->filename);

@@ -868,7 +868,7 @@ static MagickStatusType ReadPSDChannelRLE(Image \*image,const PSDInfo \*psd\_info,

" layer data is RLE compressed");

row\_size=GetPSDRowSize(image);

- pixels=(unsigned char \*) AcquireQuantumMemory(row\_size,sizeof(\*pixels));

+ pixels=(unsigned char \*) AcquireQuantumMemory(8\*row\_size,sizeof(\*pixels));

if (pixels == (unsigned char \*) NULL)

ThrowBinaryException(ResourceLimitError,"MemoryAllocationFailed",

image->filename);

diff --git a/coders/xpm.c b/coders/xpm.c

index 98e95ce..0faaf2d 100644

--- a/coders/xpm.c

+++ b/coders/xpm.c

@@ -166,7 +166,7 @@ static size\_t CopyXPMColor(char \*destination,const char \*source,size\_t length)

static char \*NextXPMLine(char \*p)

{

- assert(p != (char\*)NULL);

+ assert(p != (char \*)NULL);

p=strchr(p,'\n');

if (p != (char \*) NULL)

p++;

@@ -223,24 +223,21 @@ static char \*ParseXPMColor(char \*color,MagickBooleanType search\_start)

}

return((char \*) NULL);

}

- else

+ for (p=color+1; \*p != '\0'; p++)

+ {

+ if (\*p == '\n')

+ break;

+ if (isspace((int) ((unsigned char) (\*(p-1)))) == 0)

+ continue;

+ if (isspace((int) ((unsigned char) (\*p))) != 0)

+ continue;

+ for (i=0; i < NumberTargets; i++)

{

- for (p=color+1; \*p != '\0'; p++)

- {

- if (\*p == '\n')

- break;

- if (isspace((int) ((unsigned char) (\*(p-1)))) == 0)

- continue;

- if (isspace((int) ((unsigned char) (\*p))) != 0)

- continue;

- for (i=0; i < NumberTargets; i++)

- {

- if (\*p == \*targets[i] && \*(p+1) == \*(targets[i]+1))

- return(p);

- }

- }

- return(p);

+ if ((\*p == \*targets[i]) && (\*(p+1) == \*(targets[i]+1)))

+ return(p);

}

+ }

+ return(p);

}