Ainevsia update CVEIDs ... on Mar 27 History

..	
1.png	last year
2.png	last year
3.png	last year
4.png	last year
README.md	8 months ago

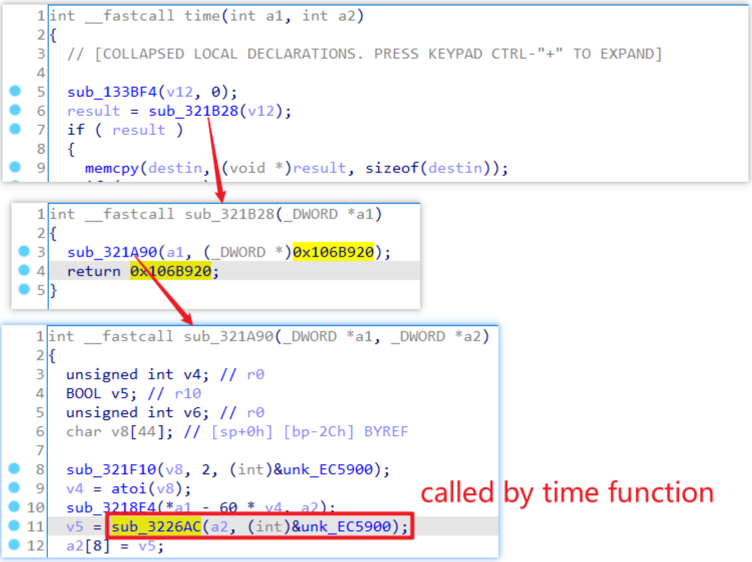
README.md

Xerox Phaser 4622 Vulnerability

This vulnerability lies in the `time` utility which influences the lastest version of Xerox Phaser 4622. The lastest version of this product is Phaser 4622 Firmware Release V35.013.01.000, according to their official website.

Vulnerability description

There is a stack buffer overflow vulnerability in function `sub_3226AC` , which is call by `time` function, as show in the figure below.

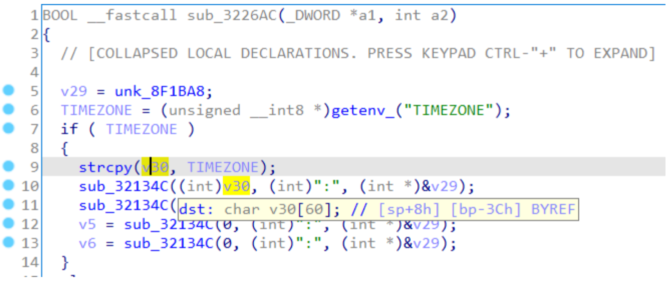


```
int __fastcall time(int a1, int a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  sub_133BF4(v12, 0);
  result = sub_321B28(v12);
  if ( result )
  {
    memcpy(destin, (void *)result, sizeof(destin));
```

```
int __fastcall sub_321B28(_DWORD *a1)
{
  sub_321A90(a1, (_DWORD *)0x106B920);
  return 0x106B920;
}
```

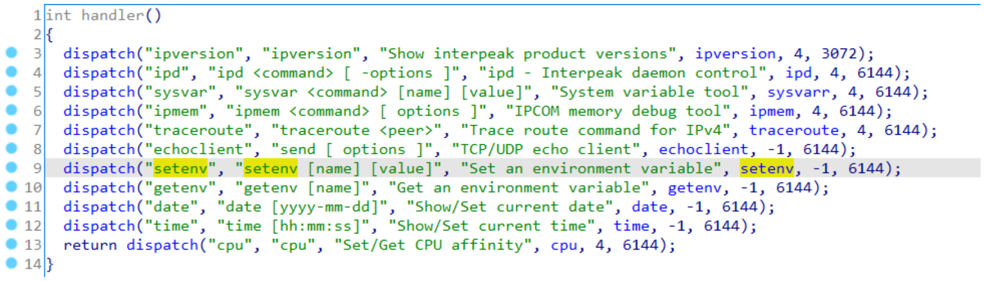
```
int __fastcall sub_321A90(_DWORD *a1, _DWORD *a2)
{
  unsigned int v4; // r0
  BOOL v5; // r10
  unsigned int v6; // r0
  char v8[44]; // [sp+0h] [bp-2Ch] BYREF

  sub_321F10(v8, 2, (int)&unk_EC5900);
  v4 = atoi(v8);
  sub_3218F4(*a1 - 60 * v4, a2);
  v5 = sub_3226AC(a2, (int)&unk_EC5900);
  a2[8] = v5;
```

The function `sub_3226AC` uses `strcpy` to copy the string pointed by `TIMEZONE` into a stack buffer pointed by `v30` . The `TIMEZONE` variable is a environment vaiable of the same name, which is accuired by function `getenv_` .

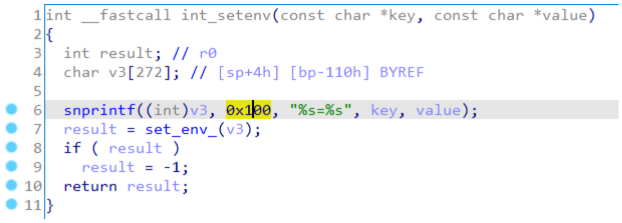


```
BOOL __fastcall sub_3226AC(_DWORD *a1, int a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v29 = unk_8F1BA8;
  TIMEZONE = (unsigned __int8 *)getenv_("TIMEZONE");
  if ( TIMEZONE )
  {
    strcpy(v30, TIMEZONE);
    sub_32134C((int)v30, (int)":", (int *)&v29);
    sub_32134C(dst: char v30[60]; // [sp+8h] [bp-3Ch] BYREF
    v5 = sub_32134C(0, (int)":", (int *)&v29);
    v6 = sub_32134C(0, (int)":", (int *)&v29);
  }
```

Any user can set any environment variable using the provided `setenv` to set any variable to any value, given that the `<key>=<value>` does not exceed `0x100` , according the the function logic. See some decompiled code snippet below.



```
int handler()
{
  dispatch("ipversion", "ipversion", "Show interpeak product versions", ipversion, 4, 3072);
  dispatch("ipd", "ipd <command> [ -options ]", "ipd - Interpeak daemon control", ipd, 4, 6144);
  dispatch("sysvar", "sysvar <command> [name] [value]", "System variable tool", sysvarr, 4, 6144);
  dispatch("ipmem", "ipmem <command> [ options ]", "IPCOM memory debug tool", ipmem, 4, 6144);
  dispatch("traceroute", "traceroute <peer>", "Trace route command for IPv4", traceroute, 4, 6144);
  dispatch("echoclient", "send [ options ]", "TCP/UDP echo client", echoclient, -1, 6144);
  dispatch("setenv", "setenv [name] [value]", "Set an environment variable", setenv, -1, 6144);
  dispatch("getenv", "getenv [name]", "Get an environment variable", getenv, -1, 6144);
  dispatch("date", "date [yyyy-mm-dd]", "Show/Set current date", date, -1, 6144);
  dispatch("time", "time [hh:mm:ss]", "Show/Set current time", time, -1, 6144);
  return dispatch("cpu", "cpu", "Set/Get CPU affinity", cpu, 4, 6144);
}
```



```
int __fastcall int_setenv(const char *key, const char *value)
{
  int result; // r0
  char v3[272]; // [sp+4h] [bp-110h] BYREF

  snprintf((int)v3, 0x100, "%s=%s", key, value);
  result = set_env_(v3);
  if ( result )
    result = -1;
  return result;
}
```

A string of length `0x100` can of course smash the stack of `sub_3226AC` .

So by first setting the `TIMEZONE` and then invoking the command line utility `time` , the attacker can easily perform a **Deny of Service Attack** or **Remote Code Execution** with carefully crafted overflow data.

POC

```
TIMEZONE=zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
```

Timeline

- 2021.07.18 report to Xerox, CVE and CNVD
- 2021.08.31 CNVD ID assigned: CNVD-2021-57348
- 2022.02.16 CVE ID assigned: CVE-2021-37354

Acknowledgment

Credit to @Ainevsia, @peanuts and @cpegg from Shanghai Jiao Tong University and TIANGONG Team of Legendsec at Qi'anxin Group.