Distribution of Remcos RAT Disguised as Payslip

By ATCP

Published: 2023-10-25 · Archived: 2026-04-05 21:16:15 UTC

AhnLab Security Emergency response Center (ASEC) has discovered circumstances of the Remcos remote control malware being distributed through an email disguised as a payslip. As shown in **Figure 1**, the identified Remcos RAT was distributed under an email subject that read ‘This is a confirmation document for your payment transfer’, deceiving the readers. The attached compressed cab file contains an EXE file (Remcos RAT) disguised with a PDF file icon as shown in **Figure 2**.

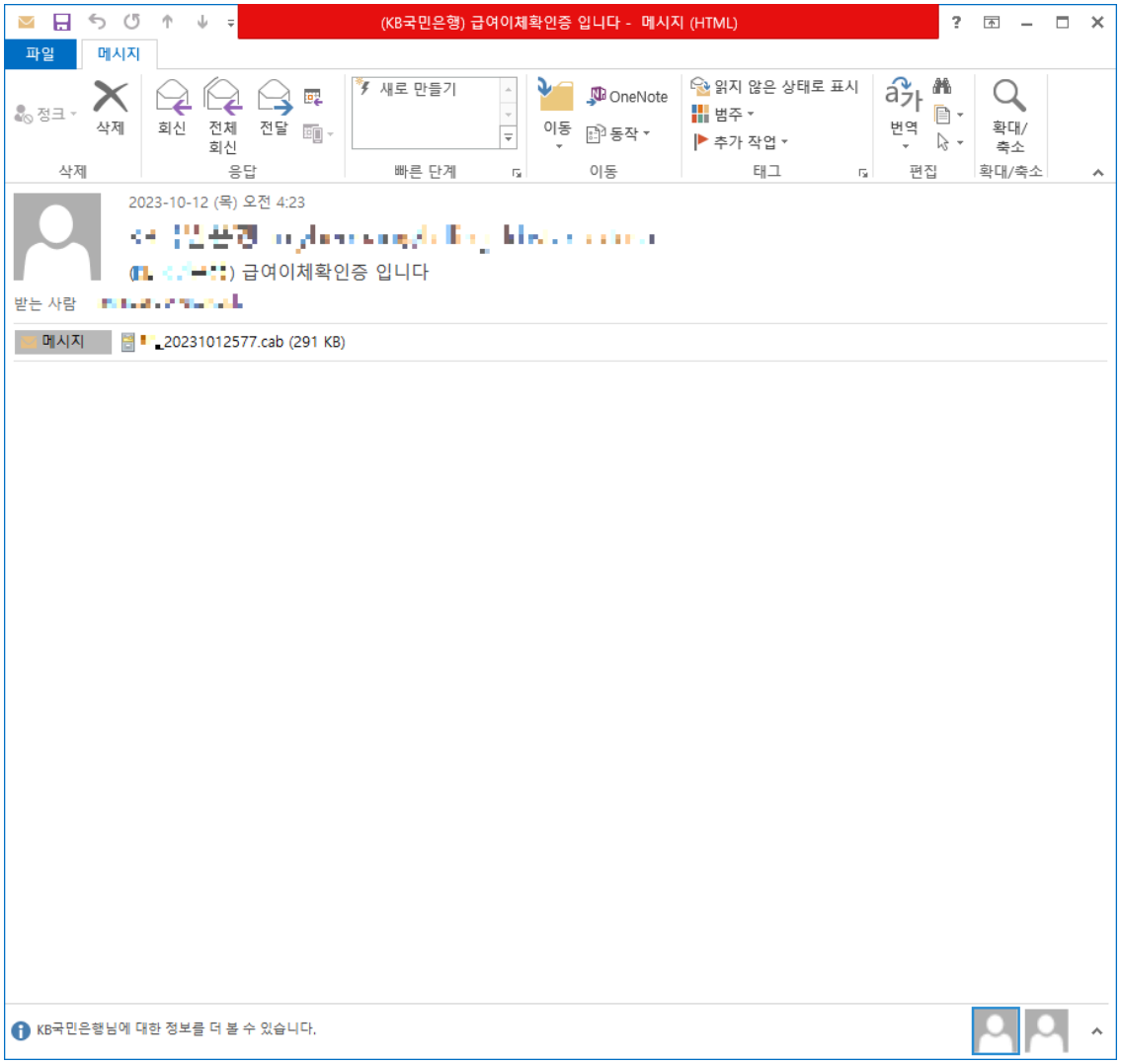

Figure 1. Phishing email

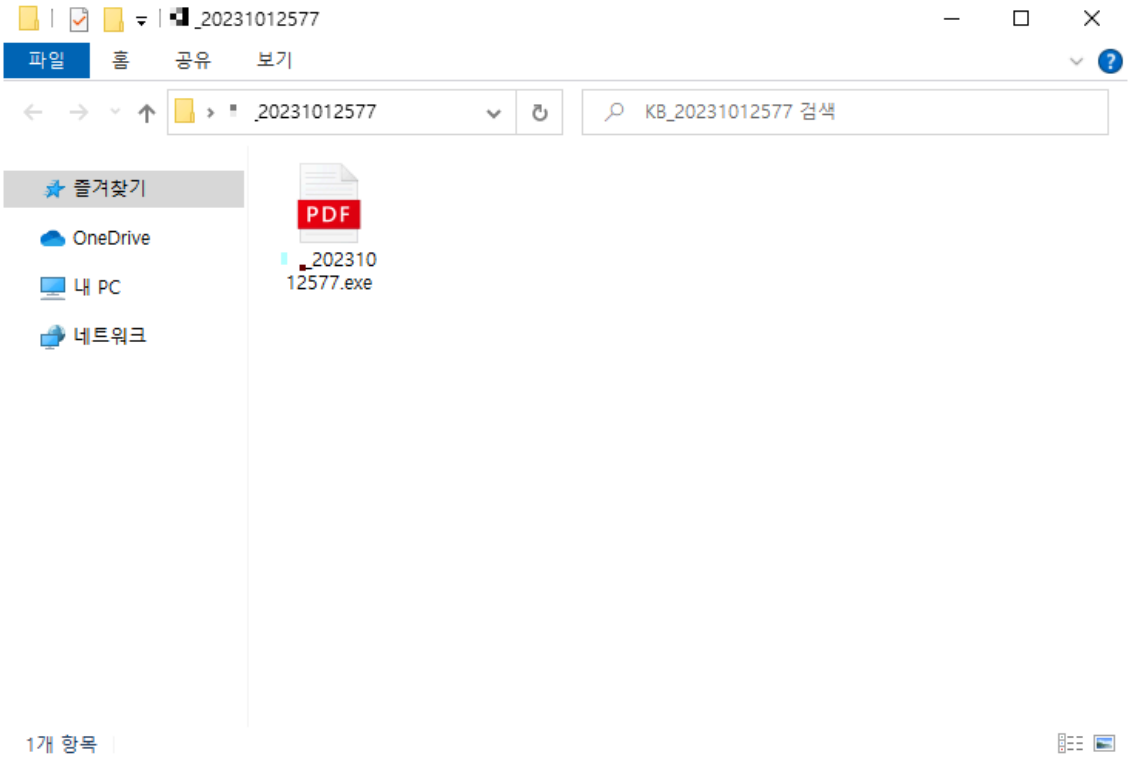

Figure 2. Remcos RAT (.exe) within the attached compressed .cab file

As shown in **Figure 3**, Remcos RAT can not only perform keylogging, screenshot capturing, and controlling webcams and microphones according to the threat actor's commands, but also enable malicious remote control such as extracting the histories and passwords saved to web browsers within the system it is installed in. **[1]** As Remcos RAT is designed for remote control, it does not exhibit any malicious behaviors until commands are received from the threat actor's server (C2). However, due to the behaviors of the offline keylogger which runs immediately after infection without any command from the C2, it can be detected with sandbox devices.

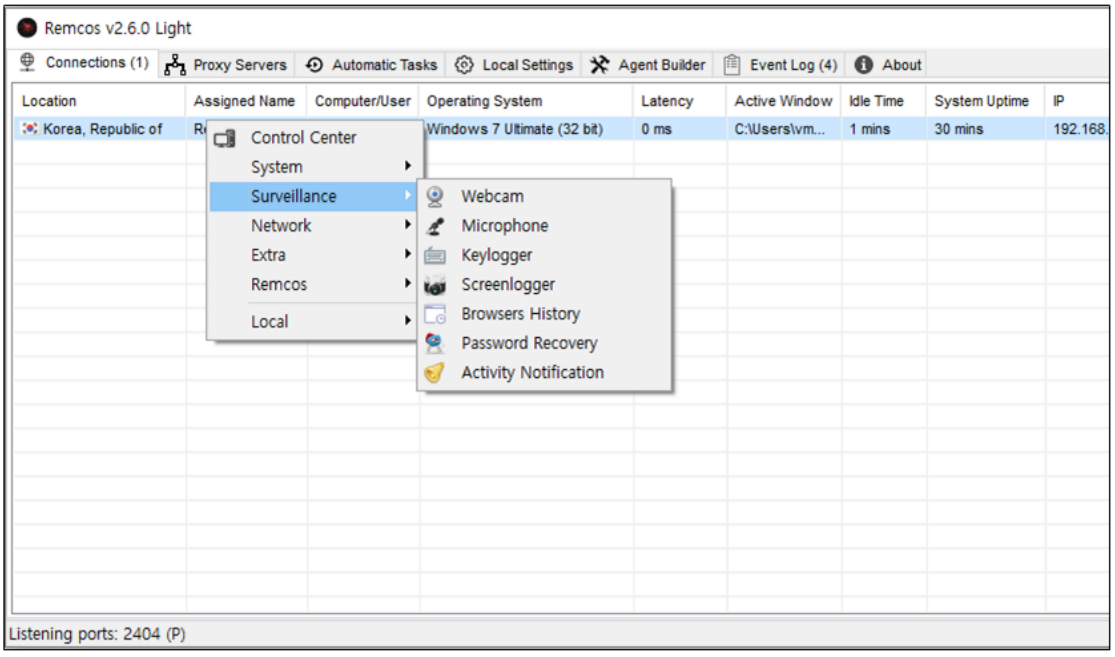

Figure 3. Various control features of the Remcos RAT's remote control server (Remcos v2.6.0)

Figure 4 shows the Remcos RAT's offline keylogger feature's functions that run without any command from the C2. Specifically, it uses the SetWindowHookExA API and installs a hook procedure that monitors keyboard input events through the WH_KEYBOARD_LL argument, as shown in **Figure 5**.

```
int __thiscall Start_Keylogger(_BYTE *lpParameter, char a2, int a3, int a4, int a5, int a6, int a7)
{
  int v9; // [esp-30h] [ebp-58h] BYREF
  char v10; // [esp-18h] [ebp-40h] BYREF
  int v11; // [esp-14h] [ebp-3Ch]
  int v12; // [esp-10h] [ebp-38h]
  int v13; // [esp-Ch] [ebp-34h]
  int v14; // [esp-8h] [ebp-30h]
  char *v15; // [esp-4h] [ebp-2Ch]
  char v16[24]; // [esp+10h] [ebp-18h] BYREF

  v15 = &a2;
  lpParameter[73] = 1;
  sub_123A83C(lpParameter + 96, v15);
  if ( byte_129E9C4 != 50 )
  {
    sub_1232073(v16, "Offline Keylogger Started");
    sub_1249BCA((int)&v10, (int)v16);
    sub_123A0B0(v10, v11, v12, v13, v14, (int)v15);
    sub_1231FB8();
  }
  sub_1232073(&v10, "Offline Keylogger Started");
  sub_1232073(&v9, "i");
  sub_12494DA();
  CreateThread(0, 0, hThread, lpParameter, 0, 0);
  if ( !*(_DWORD *)lpParameter )
    CreateThread(0, 0, SetWindowHookExA_WH_KEYBOARD_LL_13, lpParameter, 0, 0);
  CreateThread(0, 0, sub_1239311, lpParameter, 0, 0);
  return sub_1231EE9(&a2);
}
```

Figure 4. Remcos RAT's offline keylogger feature

```
int __thiscall SetWindowHookExA_WH_KEYBOARD_LL(HHOOK *this)
{
  HMODULE ModuleHandleA; // eax
  HHOOK v3; // eax
  DWORD LastError; // eax
  int v5; // eax
  int v7; // [esp-30h] [ebp-70h] BYREF
  int v8; // [esp-18h] [ebp-58h] BYREF
  char v9[24]; // [esp+Ch] [ebp-34h] BYREF
  struct tagMSG Msg; // [esp+24h] [ebp-1Ch] BYREF

  dword_129FAF4 = (int)this;
  if ( *this
    || (ModuleHandleA = GetModuleHandleA(0), v3 = SetWindowsHookExA(13, fn, ModuleHandleA, 0), (*this = v3) != 0) )
  {
    do
    {
      if ( !GetMessageA(&Msg, 0, 0, 0) )
        break;
      TranslateMessage(&Msg);
      DispatchMessageA(&Msg);
    }
    while ( *this );
    return 0;
  }
  else
  {
    LastError = GetLastError();
    v5 = sub_1249B16(v9, LastError);
    sub_12352DD(&v8, "Keylogger initialization failure: error ", v5);
    sub_1232073(&v7, "E");
    sub_12494DA();
    sub_1231FB8();
    return 1;
  }
}
```

Figure 5. Remcos RAT's keyboard input hooking code (SetWindowsHookExA)

[Detection by MDS] Figure 6 is the screen that shows the detection of the aforementioned Remcos RAT's offline keylogger feature in AhnLab MDS sandbox environment. **Figure 7** shows that the malicious behavior of hooking keyboard input has been detected.

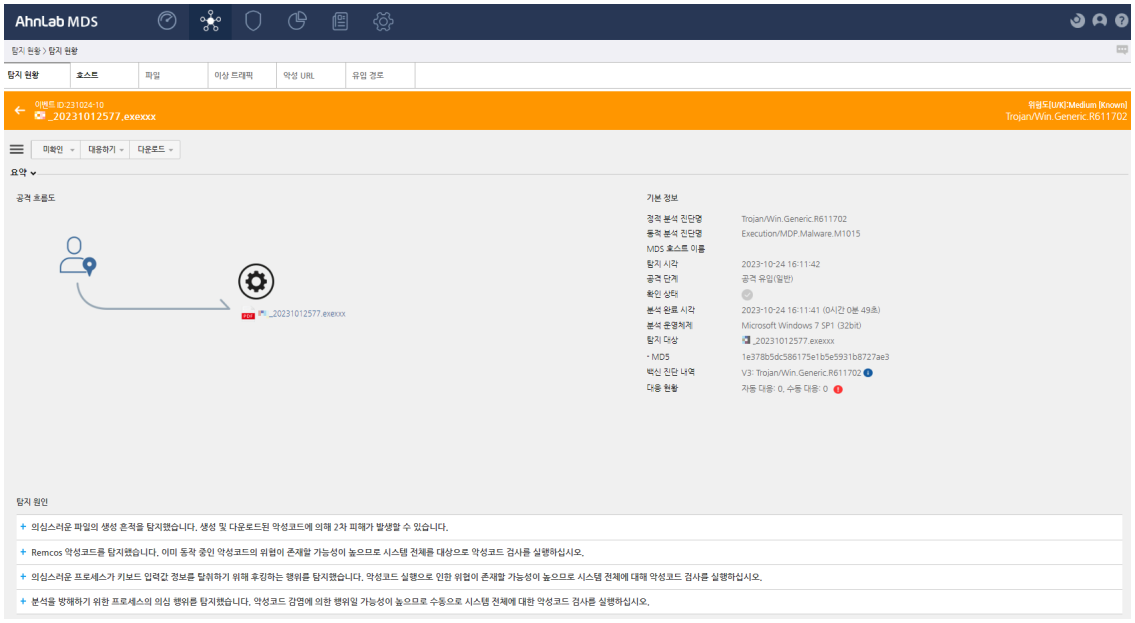

Figure 6. Remcos RAT malware detected using AhnLab MDS (1)

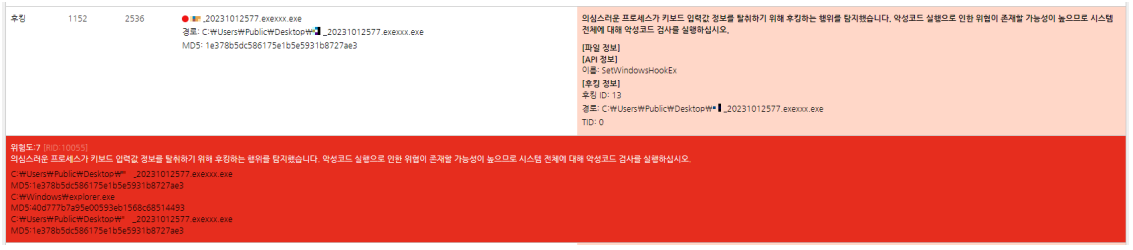

Figure 7. Remcos RAT malware detected using AhnLab MDS (2)

RAT malware performs key malicious behaviors through the commands of the threat actor. Thus, it is characteristically difficult to be aware of said infections until the threat actor's commands are run through communications with the server. To prevent security incidents and enable quick response upon breach, security administrators must not only use APT solutions such as MDS but also monitor abnormal behaviors occurring in endpoint environments with products such as EDR. **[File Detection]** – Trojan/Win.Generic.R611702 (2023.10.14.00) **[Behavior Detection]** – SystemManipulation/MDP.Hooking.M10055 – Execution/MDP.Remcos.M11099 – DefenseEvasion/MDP.AntiAnalysis.M912

MD5

1e378b5dc586175e1b5e5931b8727ae3

Additional IOCs are available on AhnLab TIP.

To learn more about **AhnLab MDS**'s sandbox-based behavioral analysis, please click the banner below.

Source: https://asec.ahnlab.com/en/58195/