Let's Learn: Deeper Dive into Ramnit Banker "VNC IFSB" Remote Control Module

vkremez.com/2018/02/deeper-dive-into-ramnit-banker-vnc-ifsb.html

Goal: Analyze Ramnit's hidden Virtual Network Computing (hVNC) remote control module focusing on its hidden desktop creation.
Source:

- Ramnit main loader (5ae2ad8f0be144ce732badf7dec0a16e)
- hVNC module (5AE2AD8F0BE144CE732BADF7DEC0A16E)
- Rig Exploit Kit landing (AS9123 TIMEWEB-AS 176[.]57[.]217[.]89)

Background:
While following the Rig Exploit Kit's distribution of the Ramnit (**demetra**) banking malware via the Seamless gate, I decided to dive deeper into its "VNC ISFB" module that is most notable running as a thread inside "TRACERT.EXE," a child process of Ramnit's svchost.exe

> 2-13-2018: #Seamless gate -> #RigEK landing
> 176.57.217[.]89 -> AS9123 TIMEWEB-AS ->#Ramnit banker #malware -> just copy/pasted code from #ISFB gang's hidden VNC module w/ hooks & "IsfbInitClient" 😂
> Hash: 5ae2ad8f0be144ce732badf7dec0a16e pic.twitter.com/NjfPBSsIXz
> — Vitali Kremez (@VK_Intel) February 13, 2018

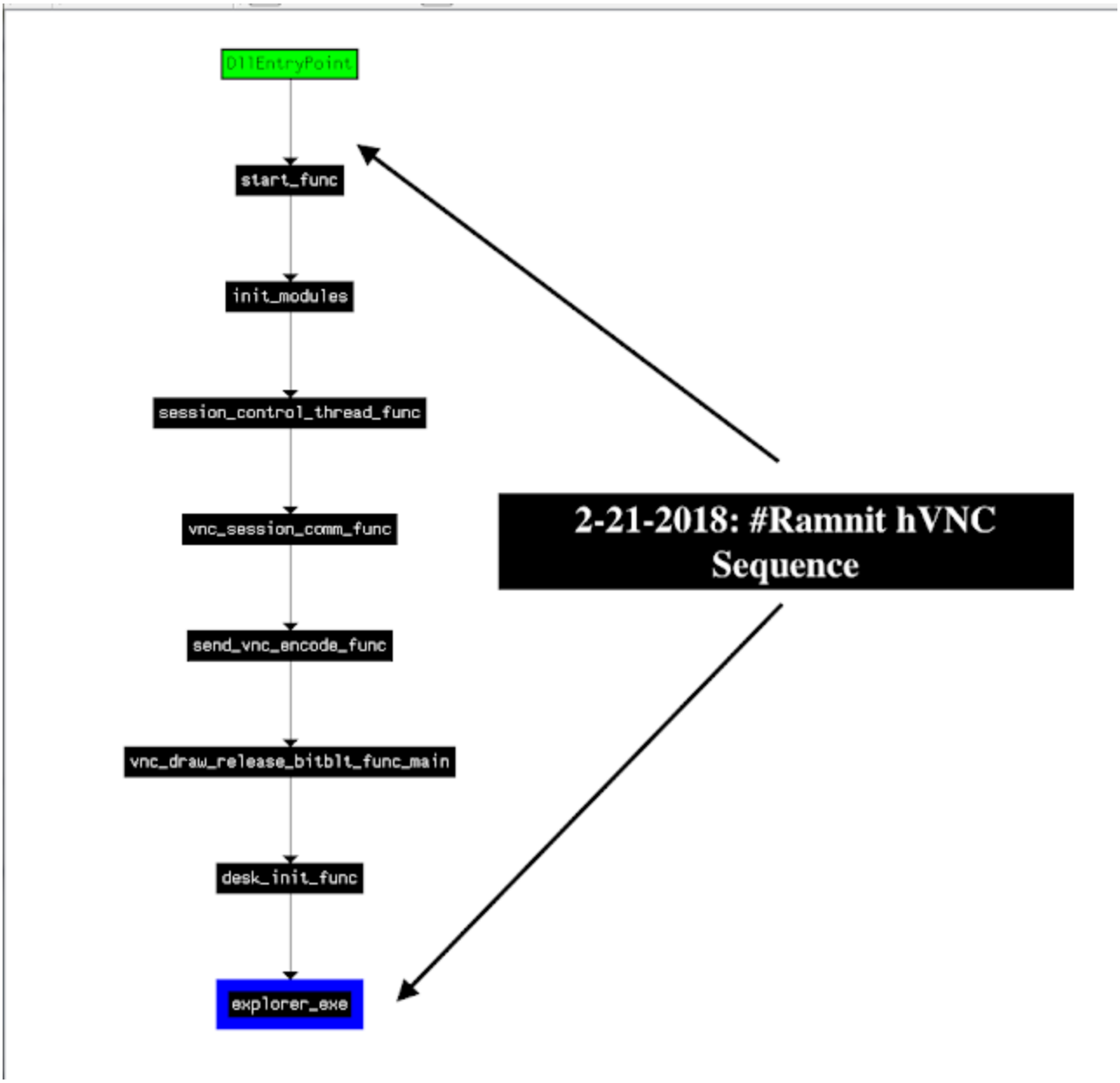
By and large, Ramnit, which is also known as "demetra" in the underground, leverages the following modules:

- Antivirus Trusted Module v2.0 (AVG, Avast, Nod32, Norton, Bitdefender)
- XX'S
- CookieGrabber
- Cookie Grabber v0.2 (no mask)
- FFCH
- FF&Chrome reinstall x64-x86 [silent]
- Hooker
- IE & Chrome & FF injector
- VNC IFSB
- VNC IFSB x64-x86

HVNC allows criminals to bypass many anti-fraud measures by allowing compromised accounts to be accessed directly from the victim's machine. By using a VNC program, which allows for remote access to and control of a machine, Ramnit actors do not have to spoof or try to replicate a victim machine's data to avoid the account being flagged. HVNC allows

actors to carry out these activities concurrent with regular user activity without being detected by operating in a hidden desktop.
The DLL module contains the following three export ordinals:

Ordinal	Export Function
00000000	PluginRegisterCallbacks
00000002	VncStartServer
00000003	VncStopServer

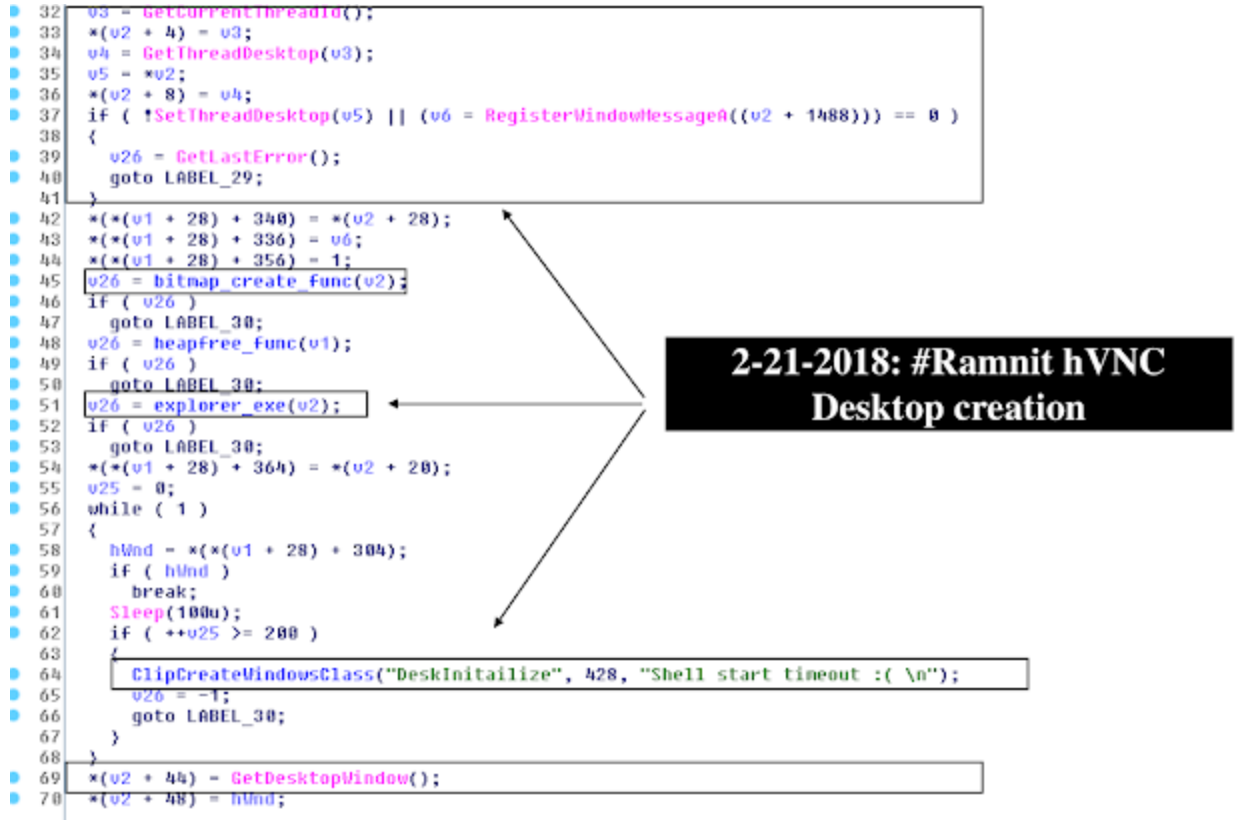


Here is one of the most interesting sequences of hidden desktop calls with injection:
Start -> **init_modules function** -> **session_control_thread function** -> **VNC session communication function** -> **send VNC encode function** -> **VNC draw BitBlt function** -> **hidden Desktop initiation** -> **creation of hidden explorer.exe** -> **create_process_inject**

function -> AcInjectDll function (check x86/x64) hooking CreateProcess* -> map memory into the process and inject it with RunPE injection

Analysis:

Ramnit's VNC module leverages a set of programs using the Remote Frame Buffer (RFB) protocol hooking multiple API calls and leveraging ISFB AcDLL injection module. Not only the whole Ramnit module was pulled directly from the ISFB gang, the module itself has strong source code similarities to the leaked Carberp one.



The following pseudocoded C++ function demonstrates the location of the "explorer.exe" with the subsequent create process injection.

```
signed int __stdcall explorer_exe(int a1)
{
  int v1;
  int v2;
  signed int result;
  CHAR *v4;
  CHAR *v5;
  int v6;
  char v7;
  int v8;
  DWORD v9;

  v1 = a1;
  v9 = 0;
  v6 = 0;
  memset(&v7, 0, 0x40u);
  v2 = GetSystemWindowsDirectoryA(0, 0);
  if ( v2 )
  {
```

```
    v4 = HeapAlloc(hHeap, 0, v2 + 15);
    v5 = v4;
    if ( v4 )
    {
      GetSystemWindowsDirectoryA(v4, v2);
      v5[v2] = 0;
      lstrcatA(v5, "\\explorer.exe");
      v8 = v1 + 1488;
      v6 = 68;
      if ( create_process_inject(v5, &v6, v1 + 12) ) // AcDLL
injection
      {
        CloseHandle(*(v1 + 16));
        CloseHandle(*(v1 + 12));
      }
      else
      {
        v9 = GetLastError();
      }
    }
    else
    {
      v9 = 8;
    }
    result = v9;
  }
  else
  {
    result = -1;
  }
  return result;
}
```

YARA RULE

```
rule crime_win32_ramnit_vnc_module_in_memory {
meta:
description = "Detects Ramnit banking malware VNC module"
author = "@VK_Intel"
reference = "Detects Ramnit VNC"
date = "2018-02-18"
hash = "888b2c614567fb5b4474ddeeb453f8cd9f44d72efb325f7e3652fd0f748c08f1"
strings:
$s0 = "Failed mapping a section to the target process, status 0x%x" fullword ascii
$s1 = "Unable to map the section into the target process, error %u" fullword ascii
$s2 = "Unable to resolve target process import, error %u" fullword ascii
$s3 = "No module found for the target process (%u) architecture" fullword ascii
$s4 = "A section of %u bytes mapped to the target process at 0x%p" fullword ascii
$s5 = "CreateProcessAsUserA %s->%s failed" fullword ascii
$s6 = "Dep PsSupGetProcessModules, ModCount = %d " fullword ascii
$s7 = "ActiveDll: PatchProcessMemory failed, error: %u" fullword ascii
$s8 = "CreateProcessAsUserW %S->%S failed" fullword ascii
$s9 = "AcInjectDll: GetOEP failed, error: %u" fullword ascii
$s10 = "Shared section mapped at 0x%p. Starting within VNC session process." fullword
ascii
$s11 = "CreateToolhelp32Snapshot (of processes) failed err=%lu" fullword ascii
condition:
11 of ($s*)
}
```