Engineering Development Group

UMBRAGE: StolenGoods
Version 2.0

User Manual

Rev. 1.0
14 January 2014

Classified By: 2417940
Reason: 1.4(c)
Declassify On: 20630405
Derived From: COL S-06

Change Log

Date	Revision	Change Description	Authority
02/14/14	New	Initial release	AE/RDB

Table of Contents

1. (U) Scope

(S//NF) This document is the User Manual for Stolen Goods v2.0, a persistence method for the Grasshopper installer. It provides a description of how Stolen Goods works, the payloads Stolen Goods can persist, and how to use Stolen Goods with Grasshopper.

(S//NF) Stolen Goods 2.0 is fundamentally different from Stolen Goods 1.0. This user guide will not include information about Stolen Goods 1.0. Please see the original Stolen Goods 1.0 user guide for all 1.0 information. All information contained here is in reference to Stolen Goods 2.0 only.

2. (U) Overview

2.1 (U) Stolen Goods Tool Description

(S//NF) Stolen Goods 2.0 (SG2) is a persistence module for Grasshopper based on components from 3rd party malware. The components were taken from malware known as Carberp, a suspected Russian organized crime rootkit. The source of Carberp was published online, and has allowed AED\RDB to easily steal components as needed from the malware. Most of Carberp was not used in Stolen Goods 2, specifically all the Bot net/Comms components. The persistence method, and parts of the installer, were taken and modified to fit our needs. All components taken from Carberp were carefully analyzed for hidden functionality, backdoors, vulnerabilities, etc.

(S//NF) Stolen Goods 2 maintains persistence by installing custom Initial Program Loader (IPL) code found in the Volume Boot Record (VBR, also known as the Partition Boot Record or PBR). Using a series of function hooks, SG2 is able to maintain execution along the Windows boot sequence, when at one point it loads a stub driver into the system to maintain code execution after the boot process is finished. The stub driver borrows some ideas and components from the original Carberp source, but most of the stub driver has been rewritten by RDB.

(S//NF) SG2 is able to persist two different payloads at once: a DLL payload (GH1 or Persistence Spec compliant) and a driver payload (JediMindTricks/AncientProtector specifically). The driver payload does **not//not** need to be signed at all, even on 64-bit Windows systems.

2.2 (U) Dependencies

(S//NF) SG2 requires Grasshopper for configuration and installation. This document assumes the user has a working Grasshopper build (along with all of Grasshopper's dependencies).

(S//NF) SG2 also has the ability to be launched through a kernel shellcode install module included in the latest version of ShellTerm (2.8+). This method installation is easier to configure compared to Grasshopper, but does not contain many of the safety checks

Grasshoper gives you. Therefore, the shellcode installer should only be used by someone who is comfortable and confident with its use.

2.3 (U) MD5 Hashes

(U) Hashes of the Stolen Goods v2.0 binaries

aplib32.lib	6384c4870c896f532b1f3ea206833579
aplib64.lib	0f7974ae87a23c8432a1aaf15e9c777d
GH_PM32.dll	eb22416167a53bf5557b09e79f80a756
GH_PM64.dll	b6c60af4ae0c0367e33e98477f5b6022
MemStub32-GH1.dll	e319577e9e80624b9699b01dcb97bfe0
MemStub32.dll	d625f211d19a0e75120b5c5c06aeb673
MemStub64-GH1.dll	17f0e0e890db1cc27d751fabe4c135ee
MemStub64.dll	dd77dbf610160fa72947d10507ca2d21
RabbitStew32.exe	9b6b3a0efc106fb2ca9673d3a17ae12c
RabbitStew64.exe	37ac0beca1f710e1438bc0a11eef47d7
stubdriver_Win7AMD64.sys	978d022b343eb1ead548b3ba9e26d65f
stubdriver_Win7x86.sys	d0b8b3428b00f30ef2aa1213942a751a
stubdriver_WinXPx86.sys	48e2b598eb61f0f06daa59189b3581ac
Uninstall_DLL32.dll	360ab324522d8622acff9de06cd0582f
Uninstall_DLL64.dll	1b5bfcb13ce45abf52dc50f4179986b6
Vbr.exe	0d907f1ff6866cb001c4edbf24f8a285

3. (U) Operation

3.1 **(S//NF)** ***Payload Types***

(S//NF) SG2 can persist up to 2 payloads: A driver payload (.sys) and a DLL payload (.dll). The latter can come in 2 forms:

- Standard Persistence Specification compliant DLLs
- Grasshopper-1 interface compliant DLLs

(S//NF) The driver payload MUST//MUST match the bitness of the target, and should be compatible with the target OS. No driver signing is required.

3.2 **(S//NF)** ***Grasshopper installation***

(S//NF) The following steps describe how to install the Stolen Goods persistence module to an existing Grasshopper build:

1. Navigate to the main folder of an existing Grasshopper build. This is the folder containing the Grasshopper builder scripts and other Grasshopper components.
2. There should be a “Modules” folder here. Copy the “StolenGoods2” directory to the “Modules” folder.
3. In the Grasshopper build folder (where “Modules” was located) there should also be a folder named “Payloads”. Copy the following provided folders into “Payloads”: Generic_DLL, Generic_GH1

3.3 (U) Configuration

(S//NF) Stolen Goods 2 is unique compared to normal Grasshopper modules in it's ability to persist two separate types of payloads. Because of this, SG2 has an odd build path when using Grasshopper. Please pay careful attention as misconfiguration can cause at best a bad install, and at worst a BSOD on the target box.

1. After installing SG2 (section 3.2), browse to the StolenGoods2 folder in Modules. Inside there will be the binary 'Vbr.exe'. Run it (double clicking it will be fine). It should create a file called 'ipl.asm'. This is the VBR persistence code. Each time you run Vbr.exe, it will remix the VBR persistence code. All but the first 12 bytes of ipl.asm will change each time Vbr.exe is run. This is important if you wish to create signature diversity between installers.
2. Start up the Grasshopper or Cricket installer
3. When choosing a payload type, choose the type that corresponds to the DLL payload you wish to persist.
   - You can persist a Persistence Spec DLL, or a GH1 compliant DLL
   - If you're not going to persist a DLL (driver only) then pick any option that is compatible with SG2
4. When prompted for the Binary path, **do not enter the path to the payload DLL you wish to use.** You will be prompted for that path later. Instead, you must enter the path to one of the following four DLLs: MemStub32, MemStub32-GH1, MemStub64, MemStub64-GH1. All four DLLs are in the SG2 folder you installed (<grasshopper install folder>\Modules\StolenGoods2). Pick the correct one based on the DLL you're going to persist and the target system.
   - If you're not using a GH1 DLL payload, pick MemStub32 or MemStub64 (based on target OS bitness). If you are, pick one of the GH1 DLLs
   - If you're not persisting a DLL (driver only) you can pick any DLL that matches the bitness of the target system
5. The builder will now prompt you to select a persistence method. For 64-bit target OS, there will be one SG2 entry (Win 7). For 32-bit target, there will be two SG2 entries (Win 7 and Win XP). Pick the one you want based on your target.
6. The SG2 persistence module will now take over and prompt you for a path to your target payload DLL. This is where you enter the path to the DLL you want to persist
   - If you're not persisting a DLL, just leave this blank and hit enter
7. Next, you'll be prompted for an optional command line to send to the payload DLL (Persistence Spec compliant DLLs). Enter a command line here if needed
   - If not needed, leave blank and hit enter
8. Next, you'll be prompted for the Environment Variable name to create to send the command line entered previously to a Persistence Spec compliant DLL. Enter that name here if needed
   - If not needed, leave blank and hit enter
9. Next, you'll be prompted for a path **on target** to write the stub DLL to. The stub DLL handles loading the payload DLL in-memory. The path will be created by Grasshopper if it doesn't already exist, if Grasshopper is able to (sufficient permissions at time of install)
   - If a DLL is not being persisted (driver only) leave this blank and hit enter

10. Finally, you'll be prompted for the path to the payload driver you wish to persist.
    - If you're not persisting a driver (DLL only) leave this blank and hit enter
    - **Make sure you are using a driver that is compatible with the target system. Grasshopper cannot perform any checks to determine if the payload driver you gave will be OK to run on the target system. Choosing an incompatible driver for the target system will most likely cause a Blue Screen of Death (BSOD) on the target machine.**
11. You may be prompted to change Grasshopper rules (two T/F questions). Unless you know what you want, you should just hit enter (default answer of F) for both questions.
12. (Grasshopper only) Type generate and hit enter
13. (Grasshopper only) Type build and hit enter

(S//NF) You're now done configuring a Grasshopper/Cricket build. The installers will be written to the given output folder. No matter what your target OS/Bitness is, a 64 and 32 bit version of the Grasshopper/Cricket DLL and EXE installers will be built. Choose what you need accordingly.

3.4 (S//NF) Shellterm Kernel Shellcode Installer

(S//NF) Stolen Goods 2 comes with the ability to be installed using Shellterms kernel shellcode launcher. **FULL DISCLOSURE: This installation method does not have any real safety checks, and requires the user to be very careful about using it. All payloads and stubs must be chosen by the user, and making an incompatible choice will, most likely, cause the system to crash with a BSOD either during installation or after reboot.**

(S//NF) Included in this delivery is a 'shellcode builder' folder. Inside will be several binaries. First, you must run (double clicking is fine) the Vbr.exe binary. This will mix the VBR assembly code and produce ipl.asm. Re-running Vbr.exe will remix the VBR persistence code, resulting in all but the first 12 bytes being different.

(S//NF) Next, using a command prompt, you'll need to run RabbitStew32.exe or RabbitStew64.exe. There are up to 5 different parameters that you can give that will be explained below. Remember, you must choose the correct binaries that are compatible with the target system, or you will BSOD the target

1. -ss <path to stub driver>: Required. Path to the stub driver to use. This is the driver that must be run first on the system, and takes care of kicking off running and maintaining the persistence method. **Failure to choose the correct driver stub could result in a system crash**. There are 3 provided options here that are based off the target OS. Pick one of the following:
    i. stubdriver_win7AMD64.sys
    ii. stubdriver_win7x86.sys
    iii. stubdriver_winXPx86.sys

2. -sd <path to stub DLL>: Required if persisting a DLL. Path to the stub DLL to use. There are four options to pick from, based upon the target bitness and payload DLL to persist:
    i. MemStub32.dll (Persistence spec DLLs)
    ii. MemStub64.dll (Persistence spec DLLs)
    iii. MemStub32-GH1.dll (GH1 DLLs)
    iv. MemStub64-GH1.dll (GH1 DLLs)
3. -sp <on target path>: Required if persisting a DLL. Path on target to lay down the stub DLL.
4. -pd <path to payload DLL>: Required if persisting a DLL. Path to the payload DLL to persist.
5. -ps <path to payload Driver>: Required if persisting a driver. Path to the driver to persist.

(S//NF) At least one payload required (DLL or Driver). You can persist one driver and one DLL at the same time if you wish. There are two RabbitStew executables (32 and 64 bit). Choose the EXE that reflects the target machine's bitness. For example, if building an installer for Win 7 32-bit, use RabbitStew32.exe.

(S//NF) RabbitStew will print out a bunch of diagnostic information, including all files it opened and read for use. Make sure the files selected are the ones you want. RabbitStew will write out one of two files: shellcode_AMD64.bin or shellcode_x86.bin. The former is written out by RabbitStew64, the latter by RabbitStew32. These files are what you will send to ShellTerm when invoking its kernel shellcode execution module.

(S//NF) Example (using provided kshellcode.py script): kshellcode '/home/user/shellcode_x86.bin'

(S//NF) The shellcode installer should return a number based on the result of the installation. If the installer returns 0, then installation was successful. If the installer returns a non-zero number, the installation failed. There are a lot of unique codes that can be returned. Some common ones are as follows:

- 900 – Previous installation of SG2 was detected. If you're 100% sure this is not the case, it may be possible that someone else installed a custom VBR implant on the machine. See the developer to get more details
- 102 – The machine is using a non-standard sector size (standard is 512 bytes). This can really mess up installation, so SG2 exits safely. If this occurs, you'll need to talk to the developer about a custom compilation to run on this machine.
- 103 – The total size of the payloads is too big for the free space found on the target machine. Typically using a DLL and Driver payload combination that totals more than 1 MB will be too big. The goal is to remain under 850 KB in total payload sizes (overhead of required components is typically < 150 KB), which would result in ~1MB total size. Space requirements are dependent on

the target machine. Some machines can have 6-7 MB of unpartitioned space, others may only have 1 MB (contiguous).

- 104 – The installer failed to write the stub DLL to disk. The likely cause of this is that the target path exists already. Example: Tried to lay down stub DLL to c:\windows\system32\msxml6.dll but msxml6.dll already exists in that directory.
- 6 – The active partition is not an NTFS partition. Not safe to install in this case.

(S//NF) Other return errors for the shellcode installer, and for the Grasshopper installer, describe many other issues that could arise. If you get a code not listed above, the developer can look it up for you.

3.5 (U) Installation Confirmation

(S//NF) To confirm an installation of Stolen Goods 2 which persisted a DLL payload, simply check that the stub DLL was laid down on disk at the target location.

(S//NF) To confirm that a driver-only payload installation of SG2 worked, you'll have to rely on the return codes of the installers. Driver-only installs write nothing to the file system on disk, and verification of installation would require inspection of the disk using something like WinHex. You could try to re-install again and see if you get a previous install return code (900 for the shellcode installer).

4. (U) Uninstall

(S//NF) If a GH1 payload is being persisted, the GH1 payload can trigger an uninstall. Otherwise, two DLLs have been provided to trigger an uninstall event. The DLLs Uninstall32 and Uninstall64 are simple Fire and Forget DLLs which tell SG2 to uninstall. A zero return value indicates successful uninstall. Non-zero represents an error. If a non-zero error value is returned, the developer look up and tell you the issue when given the error code. SG2 prevents double uninstalls, so there is no penalty in accidentally running the uninstall DLL more than once.

(S//NF) If a payload DLL was used, the stub DLL on disk will be queued for deletion after the next reboot. All other artifacts (stub driver, payload driver, payload DLL) are wiped from disk during the uninstall process (3x overwrite with zeros).

(S//NF) **Note:** GH1 uninstalls can take some time. During testing, it was possible to trigger the GH1 uninstall through something like IcePick, and restart the system before IcePick had a chance to notify the SG2 stub to uninstall. Typically, the GH1 uninstall will take 30-60 seconds to complete.

5. (U) Additional Information

5.1 (U) System Presence

(S//NF) Stolen Goods 2 will drop at most 1 file to disk (stub DLL). The stub DLL contains open-source memory load code, and code to initialize a GH1 payload (if used). It also contains code to contact the stub driver for downloading the decrypted DLL payload, and for triggering an uninstall.

(S//NF) Stolen Goods 2 saves the stub driver, payload driver (if any) and payload DLL (if any) in free space on the disk. Usually this space is between the MBR and partition entries, or in unpartitioned space at the end of the disk. The stub driver is XOR obfuscated. The payload driver and payload DLL are encrypted with a host-key that is based off information in the Bios Partition Block in the partition block. If this host-key information is changed, decryption will fail and SG2 will uninstall immediately.

(S//NF) If SG2 is installed through the shellcode installer, the payload are XOR obfuscated upon initial installation. After the first reboot, SG2 will figure out that the payloads are only XOR obfuscated, and will rewrite them to disk encrypted. Therefore, after the first reboot, all the payloads will be encrypted, regardless of install method.

(S//NF) SG2 will create registry keys for the NULL driver for use with JediMindTricks. If the payload driver is not JediMindTricks, the registry creation will still occur, and will cause no side effects on the system. The registry keys are not out of the ordinary; they are standard registry keys/values needed for a filter driver. The values do not contain file names or other information that relates to JediMindTricks, SG2, or any paths used by those tools. The keys are created for the NULL service/driver entry.

(S//NF) During the uninstall process, SG2 will write a registry key to schedule the disk stub for deletion after the next reboot. This key will be removed by Windows after reboot.

5.2 (U) Troubleshooting

(S//NF) **Q: The Grasshopper/Cricket installer blew up on me when I tried to generate/build a binary (after answering all the SG2 questions**)
(S//NF) A: Make sure all paths to binaries you want to use on your box are valid and the files exist. Typically the configuration tool will blow up if a file you gave it doesn't exist when the config tool goes to find and read the file.

(S//NF) **Q: Does this work on Windows 8?**
(S//NF) A: No. Do not use this on Windows 8. The box will be **unbootable** if you install SG2 on Windows 8/8.1. This likely applies to Windows Server 2012 as well.

(S//NF) Q: **This works on Win 7/Win XP, does it work on Windows Server 20XX?**
(S//NF) A: I do not know. SG2 will likely work just fine on Server 2003. It is untested on any OS outside of Win XP/Win 7/Win 8.X (does not work on 8.x). Chances are if it's a Server version that is similar to Win XP or Win 7, there's a good chance it will work, and if it's a server version that is similar to Win 8, it probably wont work. You should

test any new OS with SG2 before deployment, because the consequences of failure typically are BSODs or constant system boot failures.

5.3 (S) PSP characterization

5.3.1 (S//NF) Kaspersky

(S//NF) Nothing to note

5.3.2 (S//NF) 360safe

(S//NF) Nothing to note

5.3.3 (S//NF) Symantec

(S//NF) Nothing to note

5.3.4 (S//NF) Other PSPs

(S//NF) Testing should be done with PSPs not listed on the requirement. There is no reason why SG2 would not work on systems running other PSPs.

6. (U) Acronyms and Abbreviations

(U) The following acronyms and abbreviations are used in this document:

(S//NF) GH1 – Grasshopper-1 Interface

(S//NF) FNF, F&F – Fire and Forget (specification)

(S//NF) SG2 – Stolen Goods 2 (this tool).