MALWARE ANALYSIS 1

THE CURSE OF NECURS, PART 1

Peter Ferrie
Microsoft, USA

The Necurs rootkit is composed of a kernel-mode driver and a user-mode component. The rootkit makes use of some very powerful techniques, but fortunately it also has some chinks in its armour.

DRIVER ENTRY

The rootkit begins by reading the module name fields directly from an undocumented structure, instead of calling the AuxKlibQueryModuleInformation() function. It also alters the driver's size of image directly in the undocumented structure, but the purpose of this change is not known. If the module name is a filename only, because it has been loaded directly from the 'system32\drivers' directory, then the rootkit prepends '\SystemRoot\System32\Drivers\' to the name, allocates a block of memory to hold the result, and then copies the string to the memory block. Otherwise, it simply allocates a block of memory to hold the name, and then copies the name to the memory block. The rootkit allocates another block of memory to hold a copy of the registry path.

The rootkit queries the '<registry path>\DisplayName' registry value, and saves the result for use later. A previous version of the rootkit performed this query only on dates prior to 2011/11/01. It is not known why the date check existed. The rootkit queries the '<registry path>\ErrorControl' registry value, and intends to require the result to be set to zero, but in fact it continues executing even if the value is missing. This behaviour appears to be a bug, though a relatively harmless one. The rootkit queries the '<registry path>\Type' registry value, and requires the result to be set to one. It queries the '<registry path>\Start' registry value, and intends to require the result to be set to zero, but in fact it continues executing even if the value is missing. Again, this appears to be a bug. The rootkit queries the '<registry path>\Tag' registry value, and requires the result to be set to one.

It also queries the '<registry path>\ImagePath' registry value. If the ImagePath begins with '\SystemRoot\System32\Drivers\', then the rootkit checks whether that substring matches the beginning of the module path. This is how it determines whether the driver was started from that location. If the driver was started from the 'drivers' directory, then the rootkit queries the '<registry path>\group' registry value, and then checks if the group is 'Boot Bus Extender'. This is how it determines whether the driver is running as a boot-time driver.

STANDARD DRIVER

If the rootkit is not running as a boot-time driver, then it

constructs a new driver name by concatenating two random numbers, and converting the result to a string. A previous version of the rootkit used the QueryPerformanceCounter() function to acquire the initial seed, and the RtlRandom() function to generate the random number. There are multiple issues with this approach, including errors because of IRQ level, and predictable values if the performance counter service is disabled. These issues are the most likely reason why the newer version of the rootkit uses a different method to generate the random numbers: the current technique is a multiply-with-carry Random Number Generator. The Random Number Generator even uses the same values (x=123456789, y=362436069, z=77465321, c=13579 and t=916905990) as were shown when the algorithm was published in 2003. The generator is seeded with all 64 bits of the value that is returned by the 'rdtsc' CPU instruction.

Once the name has been created, the rootkit creates a new registry key under '\REGISTRY\MACHINE\SYSTEM\CurrentControlSet\Services' with that name. The rootkit then enumerates all of the registry keys under '\REGISTRY\MACHINE\SYSTEM\CurrentControlSet\Services'. It queries each key for the 'Group' registry value, watching for a reference to the 'Boot Bus Extender' group. For each registry key which describes a member of the 'Boot Bus Extender' group, which also has a 'Tag' registry value, the rootkit reads the 'Tag' registry value, increments the ID in its data, and then writes the value back to the registry. The rootkit wants to ensure that no other driver has a Tag value of one. This is explained further below.

The rootkit then sets the 'ImagePath' registry value to '\SystemRoot\System32\Drivers\<random numbers>.sys', sets the 'Group' registry value to 'Boot Bus Extender', sets the 'ErrorControl' registry value to zero (ignore all errors, and display no warnings even if the driver fails to load or initialize properly), sets the 'Type' registry value to one (kernel-mode driver), sets the 'Start' registry value to zero (automatic start), and sets the 'Tag' registry value to one.

TAG, YOU'RE IT

A likely reason why the rootkit uses the hard-coded value of one for the 'Tag' is that its author assumes (incorrectly) that drivers are loaded by *Windows* according to Tag order. In fact, drivers are gathered first according to their group, then ordered by their tag value (if it exists), and then in enumeration order for whatever remains (if the tag value doesn't exist). The group order is determined by the 'List' registry value under the '\REGISTRY\MACHINE\SYSTEM\CurrentControlSet\Control\ServiceGroupOrder' key. This list is a text string naming each of the groups in their load order. The 'Boot Bus Extended' group is usually early in the list (shortly after 'System Reserved'), but this is not a requirement. The list members are described in individual values under the '\REGISTRY\MACHINE\SYSTEM\CurrentControlSet\Control\GroupOrderList' registry key. Each value is a list of DWORDs. The first entry in the list is a count of the list subentries. Following it is an array of tags in their explicit order to be loaded. A 'Boot Bus Extended' group might be something like '6, 1, 2, 3, 4, 5, 6'. This means six entries, loading in increasing order, beginning with Tag value '1'. On the other hand, the 'SCSI Class' group might be '2, 2, 1'. This means two entries, loading Tag 2 before Tag 1. However, there is no requirement for the numbers to be sequential, and there is nothing stopping a driver from inserting itself into an arbitrary position. For example, such a driver could use tag 99 and place itself third in the list, such that the list appears '7, 1, 2, 99, 3, 4, 5, 6'. There is also nothing preventing two drivers from having the same tag value. In that case, they will be loaded in enumeration order when their tag number is requested.

The rootkit's act of increasing the tag number also introduces a potential incompatibility: since the 'Boot Bus Extended' entry in the GroupOrderList is not updated with the new tag numbers, any driver which previously had an unreferenced tag number might now be referenced explicitly, and thus load earlier than before. Conversely, any driver which previously had a referenced tag number might now be unreferenced and thus load much later than before (the most likely case is that the driver with the largest tag number, which might have loaded first – as in the 'SCSI Class' case – will now load last).

The rootkit sets the 'DisplayName' registry value either to the value that was retrieved earlier (in the case of the current version of the rootkit) or to an empty string (in the previous version of the rootkit) if the registry value was not queried.

YOU ARE UNDER MY CONTROL

If everything is successful, then the rootkit copies itself to '\SystemRoot\System32\Drivers\<random numbers>.sys'. It enumerates registry keys under the '\REGISTRY\MACHINE\SYSTEM' key to find the ones that begin with 'ControlSet' (that is, 'ControlSet001' and 'ControlSet002', by default, though there can be others). Within each of the 'ControlSet' registry keys that are found, the rootkit finds and deletes any reference to the 'Services\<random numbers>' registry key. The rootkit wants to remove references to itself from the backup of the registry, so that it does not have to hide those values.

At this point, the rootkit loads the driver from its new location, deletes the original file and the registry key that launched it, and then exits.

In part 2, we will look at what the driver does when it is loaded as a boot-time driver.