Deactivate the Rootkit: Attacks on BIOS anti-theft technologies

Alfredo Ortega, Anibal Sacco, Core Security Technologies

July 24, 2009

Contents

Abstract

This is a report on our research into anti-theft technologies utilized in the PC BIOS. In particular, we have analyzed the Computrace BIOS agent and documented some design vulnerabilities that allow the agents reporting address to be controlled. Additionally, we outline an experimental method for re-setting the permanent activation/deactivation capability of the persistent agent in the BIOS to the default factory settings. We are certain that this available control of the anti-theft agent allows a highly dangerous form of BIOS-enhanced rootkit that can bypass all chipset or installation restrictions and reutilize many existing features offered in this kind of software.

1 Introduction

Computer-based anti-theft technologies are used to prevent or deter the unauthorized appropriation of a physical system.

Embedded into the BIOS of most notebooks sold since 2005, when anti-theft technology vendors Phoenix[2] and Absolute[1] reached a licensing agreement[3] they are extremely popular today. According to the vendors own corporate fact sheet[4], Phoenix is the dominant leader in the market for portable BIOS, with 60% of all sales , and BIOS offers a high level of persistence, making it the ideal place for anti-theft technologies to reside.

The system works by periodically reporting back to a central authority. In the event of theft, the central authority can instruct the resident agent to wipe all information as a security measure, or to track the whereabouts of the system, to help recover the stolen items via subsequent law enforcement activities[1]. In order to be an effective system, the anti-theft agent must be stealthy, must have complete control of the system, and most importantly, must be highly persistent because wiping of the whole system most often occurs in the case of theft.

This activity is also consistent with rootkit behavior, the only difference being that rootkits are generally malicious, while anti-theft technologies act as a form of protection against thieves.

However, in the course of researching matters of BIOS security we found that a lack of strong authentication in the most popular anti-theft technologies are the source of vulnerabilities that can lead to a complete and persistent compromise of an affected system, as we will explain in the rest of this article.

2 Computrace Agent

While doing the research that resulted in the publication of the 'Persistent BIOS Infection' article at the CanSecWest and Syscan Conference in early 2009, the Computrace persistent Agent was found in multiple notebook BIOS systems. Upon further investigation, a complete description of the agent was found on the United States Patent Application US 2006/0272020 A1. This information is in the public domain, and many complaints about

[1] from Absolute site: *"Absolute has partnerships with tier one PC OEMs such as Dell, Fujitsu, Gateway, HP, Lenovo, Motion Computing, Panasonic and Toshiba. Embedded in the BIOS firmware of a computer, the Computrace Agent can survive operating system re-installations, hard drive reformats and even hard drive replacements. The company has reselling partnerships that extend beyond this list of OEMs that include global leaders such as Apple and Toshiba. Absolute has also established strategic relationships with more than 1000 police departments, government security agencies and private security firms throughout North America and around the world."*

the involved technology have been posted online, some even with valid instructions for erasing Computrace completely from the BIOS [6].

The Agent in question is a PCI Option ROM embedded version found on most notebook BIOS, and some Desktop BIOS systems as well. The Optional ROM is deactivated by default as the PCI device that it refers to (1917:1234) doesn't exist. Upon activation, it modifies the underlying Windows file system *directly from the BIOS*[2], installing a new service and modifying various core system files like the registry, and self-healing mechanisms including Autochk.exe.

The Computrace anti-theft system also has the capability to read Bitlocked file systems on Windows Vista.

The BIOS Agent supports Windows 98/XP and Vista, with either FAT32 or NTFS file systems. We studied many versions of the Computrace Agent, including V80.845 and V80.866. Once installed and with Windows fully booted, the agent runs as a Windows service and proceeds to contact a remote system and wait for orders. This process can consist in the downloading of additional software or reporting of various run-time parameters.

2.1 BIOS code

The following hexadecimal dump details the Computrace PCI Option ROM header found inside the BIOS of a HP 9420 notebook computer. The Option ROM is deactivated because it correspond to the PCI Device 1917:1234, inexistent on the system.

```
00000000  55 aa 2a eb 15 43 6f 6d  70 75 54 72 61 63 65 20  |U.*..CompuTrace |
00000010  56 38 30 2e 38 36 36 78  1d 00 e9 5c 01 50 43 49  |V80.866x...\.PCI|
00000020  52 17 19 34 12 00 00 18  00 00 06 00 00 2a 00 00  |R..4.........*..|
```

Interestingly, Computrace uses the UPX packing software, version 1.00, you can see the UPX! signature near offset 0x200:

```
00000200  57 e9 45 e2 55 50 58 21  0b 01 04 09 45 78 74 75  |W.E.UPX!....Extu|
00000210  c2 ae 1a 79 58 e2 b9 4f  04 26 ed ff 8c 16 00 ff  |...yX..O.&......|
```

2.2 Detail of agent operation

When installed, the deployed agent registers itself as a normal windows service using the name "Remote Procedure Call (RPC) Net". This name, with slight variations, is also used by Windows to refer other legitimate services as "Remote Procedure Call (RPC)" (Used to provide the endpoint mapper and other RPC Services) and "Remote Procedure Call (RPC) Locator" (In charge of managing the RPC name service database). In this way, the registered service could be easily confused with these legitimate Windows services, except for its lack of a description. The service is implemented on the rpcnet.exe or rpcnetp.exe file.

[2]The Agent has small but functional file system drivers

3 Report URL redirection. Who is the thief?

The persistent agent uses a configuration method consisting in a 512-byte block of data. This block contains configuration items like IP, port and URL of report, as well as expiration date and AT commands (The agent also has modem reporting capabilities). The block can reside in many places. It's hard-coded inside the Option-ROM, with the 'search.namequery.com' URL and IP used as the default reporting point, as we show in section3.1.

However, on the first run this configuration block is copied in many places, including the registry and hard-disk inter-partition space. Allowing the agent in this way to survive hard disk formats. Again, the obfuscation method used in this configuration block is a XOR operation, this time against the 8-bit key 0xB5. The block is obfuscated on a slightly more convoluted way on the registry. But the encryption algorithm is similar, making the modification trivial.

We are presenting a method to search and modify this configuration block, pointing the IP and URL to a malicious site, where un-authenticated payloads can be directed to the involved notebook. Modification of the block in the inter-partition space allows for a format-resistant malicious agent. On unsigned BIOSes, direct Option ROM modification of the configuration block allows for a very persistent and dangerous form of rootkit, taking in account that anti-virus software will ignore the agent, recognizing it as the normal Computrace agent, as no modification to the Agent itself is being made.

3.1 Configuration block

Below is the configuration block used by the Computrace agent V80.866, it was extracted from the Option ROM with the UPX utility. The Configuration block starts at offset 0x3c38:

```
00003c30  00 00 00 00 00 00 00 00  04 02 00 00 80 1e 04 01  |................|
00003c40  00 40 00 1f 04 00 00 00  00 10 0a f4 f4 85 f8 84  |.@..............|
00003c50  ec 85 85 85 85 1d 02 00  00 46 06 00 00 00 00 00  |.........F......|
00003c60  00 47 06 00 00 00 00 00  00 48 1a b5 e5 64 80 c4  |.G.......H...d..|
00003c70  a2 c6 d0 d4 c7 d6 dd 9b  db d4 d8 d0 c4 c0 d0 c7  |................|
00003c80  cc 9b d6 da d8 0a 02 07  10 06 06 00 00 00 00 00  |................|
00003c90  00 07 06 00 00 00 00 00  00 0f 06 b6 69 ce 05 05  |............i...|
00003ca0  96 08 06 19 99 08 08 12  12 0b 02 62 03 14 04 39  |...........b...9|
00003cb0  00 80 00 20 04 00 00 00  00 15 04 00 00 00 00 19  |... ............|
00003cc0  1b 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00003cd0  00 00 00 00 00 00 00 00  00 00 00 00 1a 01 00 1b  |................|
00003ce0  06 00 00 00 00 00 2d 01  b8 2d 01 b8 33 01 b8     |......-..-..3..|
00003cf0  2b 04 f4 e1 f1 e1 28 03  00 00 00 01 38 01 e1 ed  |+.....(.....8...|
00003d00  81 b8 33 01 b8 2b 04 f4  e1 f1 e1 28 03 00 00 00  |..3..+.....(....|
00003d10  01 23 01 00 00 00 00 00  00 00 00 00 00 00 00 00  |.#..............|
00003d20  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00003e30  00 00 00 00 00 00 7f 00  00 00 00 00 00 00 00 00  |................|
```

Doing a 8-bit XOR with 0xB5, we can see the plain-text configuration:

```
00000000  b1 b7 b5 b5 35 ab b1 b4  b5 f5 b5 aa b1 b5 b5 b5  |....5...........|
00000010  b5 a5 bf 41 41 30 4d 31  59 30 30 30 30 a8 b7 b5  |...AA0M1Y0000...|
00000020  b5 f3 b3 b5 b5 b5 b5 b5  b5 f2 b3 b5 b5 b5 b5 b5  |................|
00000030  b5 fd af 00 50 d1 35 71  17 73 65 61 72 63 68 2e  |....P.5q.search.|
00000040  6e 61 6d 65 71 75 65 72  79 2e 63 6f 6d bf b7 b2  |namequery.com...|
00000050  a5 b3 b3 b5 b5 b5 b5 b5  b5 b2 b3 b5 b5 b5 b5 b5  |................|
00000060  b5 ba b3 03 dc 7b b0 b0  23 bd b3 ac 2c bd bd a7  |.....{..#...,...|
00000070  a7 be b7 d7 b6 a1 b1 8c  b5 35 b5 95 b1 b5 b5 b5  |.........5......|
```

```
00000080   b5 a0 b1 b5 b5 b5 b5 ac   ae b5 b5 b5 b5 b5 b5 b5   |................|
00000090   b5 b5 b5 b5 b5 b5 b5 b5   b5 b5 b5 b5 b5 b5 b5 b5   |................|
000000a0   b5 b5 b5 b5 af b4 b5 ae   b3 b5 b5 b5 b5 b5 b5 98   |................|
000000b0   b4 0d 98 b4 0d 86 b4 0d   9e b1 41 54 44 54 9d b6   |..........ATDT..|
000000c0   b5 b5 b5 b4 8d b4 54 58   34 0d 86 b4 0d 9e b1 41   |......TX4......A|
000000d0   54 44 54 9d b6 b5 b5 b5   b4 96 b4 b5 b5 b5 b5 b5   |TDT.............|
000000e0   b5 b5 b5 b5 b5 b5 b5 b5   b5 b5 b5 b5 b5 b5 b5 b5   |................|
000001f0   b5 b5 b5 b5 b5 b5 b5 b5   b5 b5 b5 b5 b5 b5 ca      |...............|
```

With the port clearly visible at offset 0x32, IP at offset 0x35 and URL at 0x39. The communication is made via plain HTTP connections, using wininet.dll exported functions.

This is the hard-coded block located on BIOS. When installed, the agent copies this block to the registry keys:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\rpcnet\Parameters

or

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\rpcnet\Parameters subkey *Security*, depending if the Persistent (BIOS) agent is used or not.

Unpacked, the configuration block is easily modifiable. By simply changing the URL or IP, we can redirect the agent queries to our site. This is very easy to accomplish in the registry, but we don't have persistence for merely modifying the registry. To modify the configuration of the persistent agent we need to modify and reflash the BIOS. This is possible in many systems at the date of publication for this article, as unsigned BIOS are common.

4 Computrace Agent stub: Bios code execution

As we said on section 2, we found many incarnations of the persistent agent.

One particular example , found on notebooks like Dell Vostro 1510, is the Computrace V 70.785 agent (this number may change with the BIOS version). This agent doesn't contain any code except for a small stub used to load additional code from a sector on the hard disk located outside normal partitions. This is also documented on the public patent application US 2006/027220 A1.

The code on the hard-disk contains a small header that indicates the stub where to load the code in the memory, and carry out a CRC-16 check. We found the lack of code authentication in this particular case provides an easy way to build a BIOS- rootkit attack, as an unauthorized privileged user could put code on hard disk that will be executing directly on the BIOS.

5 Factory-reset of permanent activation/deactivation

On some notebooks models like the Dell Inspiron series, the persistent agent can be permanently activated or deactivated with an option on the BIOS setup utility. This is accomplished using SMBIOS Tokens, number 0x175

and 0x176 for activation and deactivation respectively, and the configuration data is stored in NVRAM instead of CMOS.

We will show a method to reset the NVRAM via a malfunction of the SMBIOS, producing a race-condition with the Dell Client Configuration Utility (DCCU), therefore resetting the persistent agent activation status to factory defaults. This allows for an all-software activation-deactivation method, demonstrating that no permanent activation or deactivation can be achieved. Please refer to Section 9 for instructions on how to reproduce this failure in Dell Inspiron 1525 models.

6 Further information

There are other anti-theft technologies today that contains a BIOS-component, like Phoenix Failsafe and Intel Anti-theft technology[5], but no research has been conducted on those systems so far by CoreLabs.

7 Conclusion

At this time, we found three major problems with common Absolute-Computrace Implementations:

1. Lack of authentication of configuration options, leading to report redirection.
2. Lack of authentication of code in stub agent, leading to bios code execution.
3. On at least one specific setup, activation/deactivation of the Computrace Agent can be reverted to factory defaults.

For issues 1 and 2 a digital signature scheme would fix the issues. We don't have any recommendation for the issue number 3 at this moment. Furthermore, there are couple of issues that at the time of this report we can't confirm:

4 Unauthenticated code download from the Agent once activated.

5 Unauthenticated BIOS agent activation.

Issues 1, 4 and 5 combined would allow for an extremely dangerous BIOS-assisted rootkit software attack to be deployed on the majority of notebooks today.

Issue 2 is dangerous by itself, providing a simple and reliable method to execute any code in the context of the BIOS, once the Option-ROM is activated.

8 Acknowledgements

Thanks to Core Security for giving us the space and resources to work in this project.
To Anton Borisov for his excellent bios analysis tools, phnxdeco, awardeco and many others.
Finally, to the Coreboot project for his FlashRom tool, that did the heavy lifting work on our research.

9 DCCU settings

You can use the following XML file as TaskResult.xml, loading it via the DCCU web interface. The version used is DCCU 3.0.1213. Loading into DCCU and applying the settings to the BIOS will cause the malfunction that will reset the NVRAM.

```
<root>
        <command name="Inventory">
                <property name="OnBoardDevices.AGPSlot" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.BuiltinNIC" value="3" errorcode="0x0" />
                <property name="OnBoardDevices.BuiltinFloppy" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.BuiltinNIC2" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.BuiltinPointingDevice" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.IntegratedAudio" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.InternalMiniPCI" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.MediaCardAnd1394" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.Microphone" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.Onboard1394" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.OnboardModem" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.ParallelPortConfiguration" value="3" errorcode="0xFFFFFFFE" />
                <property name="OnBoardDevices.PCCard" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.PCCardAnd1394" value="3" errorcode="0x0" />
                <property name="OnBoardDevices.PCISlots" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.SmartCardReader" value="2" errorcode="0xFFFFFFFC" />
                <property name="OnBoardDevices.TabletButtons" value="2" errorcode="0xFFFFFFFC" />
                <property name="Drives.IDEController" value="3" errorcode="0x0" />
                <property name="Drives.IntegratedSATAController" value="5" errorcode="0x0" />
                <property name="USB.USBPortsExternal" value="3" errorcode="0x0" />
                <property name="USB.USBWake" value="4" errorcode="0x0" />
                <property name="MTN.ModuleBayDevice" value="3" errorcode="0x0" />
                <property name="PWR.AutoOn" value="3" errorcode="0x0" />
                <property name="PWR.AutoOnHour" value="0" errorcode="0x0" />
                <property name="PWR.AutoOnMinute" value="0" errorcode="0x0" />
                <property name="PWR.CPUVirtualization" value="4" errorcode="0x0" />
                <property name="PWR.HardDiskAcousticMode" value="3" errorcode="0x0" />
                <property name="PWR.MultiCore" value="3" errorcode="0x0" />
                <property name="PWR.SingleCoreTurboMode" value="3" errorcode="0x0" />
                <property name="PWR.SpeedStep" value="3" errorcode="0x0" />
                <property name="PWR.WakeupOnLAN" value="3" errorcode="0x0" />
                <property name="POST.ExternalHotkey" value="3" errorcode="0x0" />
                <property name="POST.FastBoot" value="4" errorcode="0x0" />
                <property name="POST.Keypad" value="2" errorcode="0x0" />
                <property name="POST.NumLock" value="3" errorcode="0x0" />
                <property name="POST.USBEmulation" value="4" errorcode="0x0" />
                <property name="Security.ChassisIntrusion" value="2" errorcode="0xFFFFFFFC" />
                <property name="Security.ChassisIntrusionStatus" value="4" errorcode="0xFFFFFFFE" />
                <property name="Security.NoExecute" value="4" errorcode="0x0" />
                <property name="Security.PasswordBypass" value="3" errorcode="0x0" />
                <property name="Security.TrustedPlatformModule" value="2" errorcode="0xFFFFFFFC" />
                <property name="Security.TrustedPlatformModuleActivation" value="2" errorcode="0xFFFFFFFC" />
                <property name="Wireless.BluetoothDevices" value="3" errorcode="0x0" />
                <property name="Wireless.CellularRadio" value="3" errorcode="0x0" />
                <property name="Wireless.RadioTransmission" value="2" errorcode="0xFFFFFFFC" />
                <property name="Wireless.WiFiCatcherChanges" value="4" errorcode="0x0" />
                <property name="Wireless.WiFiLocator" value="3" errorcode="0x0" />
                <property name="Wireless.WirelessLAN" value="3" errorcode="0x0" />
                <property name="Wireless.WirelessSwitchBluetoothControl" value="3" errorcode="0x0" />
                <property name="Wireless.WirelessSwitchCellularControl" value="3" errorcode="0x0" />
                <property name="Wireless.WirelessSwitchWirelessLANControl" value="3" errorcode="0x0" />
                <property name="SS.AssetTag" value="Hola loco" errorcode="0x0" />
                <property name="Configuration.PropertyOwnershipTag" value="Tagservice" errorcode="0x0" />
```

```
                <property name="SS.BIOSDate" value="2006-06-09T00:00:00" errorcode="0x0" />
                <property name="SS.BIOSVersion" value="XXXX" errorcode="0x0" />
                <property name="SS.SystemName" value="Test" errorcode="0x0" />
                <property name="SS.PowerMgtSupported" value="5" errorcode="0x0" />
                <property name="SS.Status" value="" errorcode="0x8004100E" />
                <property name="SS.ServiceTag" value="_____" errorcode="0x0" />
                <property name="SS.SystemDescription" value="_____" errorcode="0x0
                <property name="SS.SystemVendor" value="_____" errorcode="0x0" />
                <property name="SS.ProcessorType" value="1" errorcode="0x0" />
                <property name="SS.ProcessorSpeed" value="2666" errorcode="0x0" />
                <property name="SS.SystemClass" value="1" errorcode="0x0" />
                <property name="SS.AssetTag" value="_____" errorcode="0x0" />
                <property name="SS.ManufacturerDate" value="" errorcode="0x8004100E" />
        </command>
</root>
```

References

[1] Absolute and "Lojack for laptops" tracking software. http://www.absolute.com/

[2] Phoenix Technologies, PC BIOS software. http://www.phoenix.com/en/Home/default.htm

[3] Phoenix And Absolute Software Combine to Make PCs Secure From the Start With Built-in Asset Recovery and Tracking http://investor.phoenix.com/releasedetail.cfm?ReleaseID=155976

[4] Phoenix Corporate fact sheet http://files.shareholder.com/downloads/PTEC/0x0x212965/5527D53C-1616-4454-AD58-0A1DB039E654/FactSheet.pdf

[5] Intel Anti-Theft technology for notebooks http://www.intel.com/technology/anti-theft/index.htm

[6] How to remove Computrace Lojack http://www.freakyacres.com/remove_computrace_lojack