称为上帝模式恶意软件的 NSA BIOS 后门（第一部分：DEITYBOUNCE）

非官方中文译本•安天实验室 译注

文档信息			
原文名称	NSA BIOS Backdoor a.k.a. God Mode Malware（Part 1: DEITYBOUNCE）		
原文作者	Darmawan Salihun	**原文发布日期**	2014 年 1 月 29 日
作者简介	Darmawan Salihun 自 2002 年以来一直专注于 BIOS 相关的安全研究，他已在《破译者杂志》上出版了 BIOS 逆向工程、BIOS 代码注入技术、BIOS 黑客技术、即插即用 BIOS 和 PCI（外设部件互连标准）协议开发的多篇论文。2006 年，他撰写了《BIOS 反汇编忍术》一书，介绍了他对 BIOS 安全研究的结果。他目前仍在从事 BIOS 和 UEFI（统一的可扩展固件接口）安全研究。他还是资讯安全研究所的研究员。 请参阅文中的“作者简介”部分。		
原文发布单位	Infosec Institute（资讯安全研究所）		
原文出处	http://resources.infosecinstitute.com/nsa-bios-backdoor-god-mode-malware-deitybounce/		
译者	安天技术公益翻译组	**校对者**	安天技术公益翻译组
免责声明	• 本译文译者为安天实验室工程师，本文系出自个人兴趣在业余时间所译，本文原文来自互联网的公共方式，译者力图忠于所获得之电子版本进行翻译，但受翻译水平和技术水平所限，不能完全保证译文完全与原文含义一致，同时对所获得原文是否存在臆造、或者是否与其原始版本一致未进行可靠性验证和评价。 • 本译文对应原文所有观点亦不受本译文中任何打字、排版、印刷或翻译错误的影响。译者与安天实验室不对译文及原文中包含或引用的信息的		

称为上帝模式恶意软件的 NSA BIOS 后门

（第一部分 DEITYBOUNCE）

Darmawan Salihun

2014 年 1 月 29 日

本文是一系列 NSA BIOS 后门分析文章的第一部分。在开始之前，我想说一下为什么这些恶意软件被称为“上帝模式”。首先，大多数恶意软件使用“神”领域的词汇作为内部代号，例如 DEITYBOUNCE、GODSURGE（deity 和 god 都是指神）等。其次，这些恶意软件的能力类似于视频游戏中的“上帝模式”秘籍，使用该模式的玩家会近乎打遍天下无敌手。这类恶意软件也是如此，在其部署期间，即使使用最先进的反恶意软件工具，也非常难以检测和删除。

该系列文章的第一部分侧重于分析爱德华·斯诺登泄露的 NSA ANT 服务器文档中所描述的 DEITYBOUNCE 恶意软件。本文的分析基于该文档所提供的信息的技术影响。文档并没有提供很多技术细节，但是基于 DEITYBOUNCE 开始投入使用时的 BIOS 技术，我们可以推断出一些技术上合理的假设或结论。

DEITYBOUNCE 简介

DEITYBOUNCE 作为系统的一部分运作，如图 1 所示。图 1 展示了几个特有的术语，如 ROC、SNEAKERNET 等。有些术语是 NSA（美国国家安全局）内部使用的。ROC 是 remote operation center（远程操作中心）的缩写。ROC 是指 NSA 对目标系统的控制点；它位于 NSA 总部之外。SNEAKERNET 是一个极妙的术语，指物理数据的传输，即人们利用可移动介质，如磁带、软盘、光盘、USB 闪存驱动器（存储器，USB），或外部硬盘驱动器将一台计算机中的数据传输到另一台。

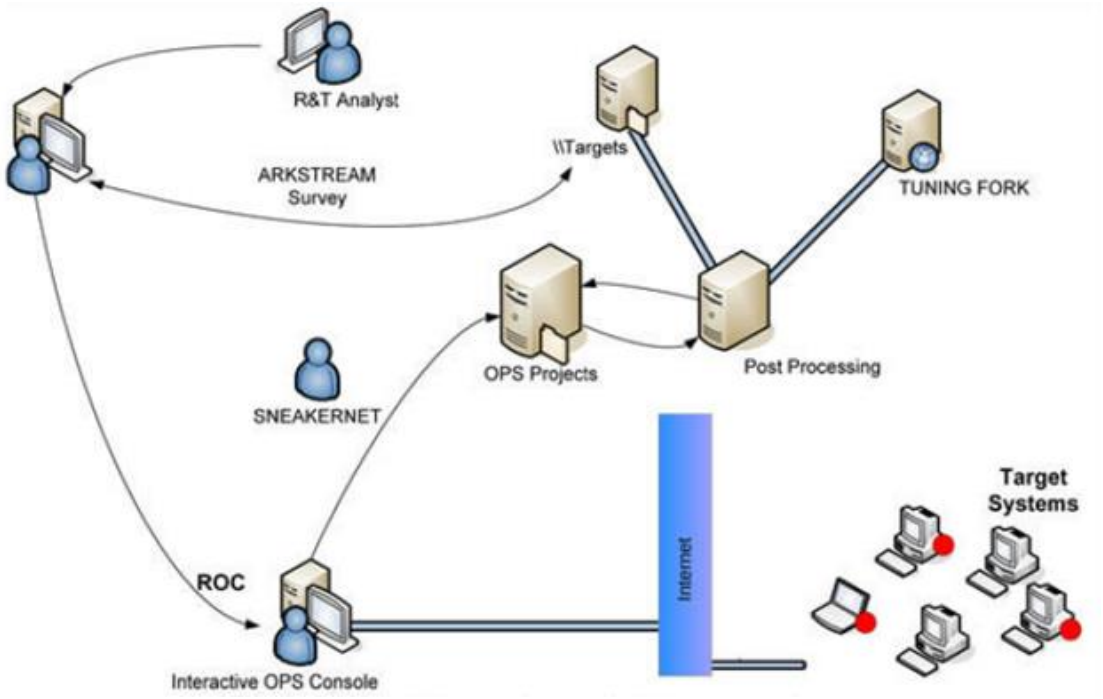

图 1：DEITYBOUNCE 扩展的操作概念

图 1 显示了 DEITYBOUNCE 针对标有红点的机器。DEITYBOUNCE 本身并不会安装在这些机器上，因为 DEITYBOUNCE 针对戴尔 PowerEdge 1850/2850/1950/2950 RAID 系列服务器（BIOS 版本 A02，A05，A06，1.1.0，1.2.0，1.3.7），而不是笔记本电脑或台式机/工作站。这意味着 DEITYBOUNCE 安装在红点标记的笔记本电脑或台式机/工作站所访问的服务器上。红点还意味着目标系统可以充当“跳转主机”，以便在目标服务器中植入 DEITYBOUNCE。

图 1 还表明了 ARKSTREAM 的存在。ARKSTREAM 基本上是一个恶意软件投放器，包含 BIOS 闪烁和恶意软件投放功能。ARKSTREAM 可以通过远程控制漏洞（网络感染）或者通过 USB 驱动器感染目标，并在目标服务器上安装 DEITYBOUNCE。在某种程度上，这种感染方法与 Stuxnet 恶意软件投放器是非常相似的。ARKSTREAM 通过 BIOS 闪烁安装 DEITYBOUNCE，即用被 DEITYBOUNCE 恶意软件“感染”的 PowerEdge 服务器 BIOS 替代正常的 BIOS。

NSA ANT 服务器文档并没有透露 DEITYBOUNCE“技术背景”的详细信息。但是，我们可以根据文件中提到的 DEITYBOUNCE 技术要求来推断一些“技术背景”。以下是 NSA ANT 服务器文档透露的重要技术细节：

1. DEITYBOUNCE 利用主板 BIOS 和 SMM（系统管理模式）持续存在于戴尔 PowerEdge

服务器上，在操作系统加载时获得定期执行。

2. DEITYBOUNCE 支持具备 RAID 硬件和 Microsoft Windows 2000、XP 和 2003 服务器的多处理程序系统。

3.一旦植入，DEITYBOUNCE 的执行频率（投放有效载荷）是可配置的，并会在目标机器启动时执行。

在后面的章节中，我们将基于这些技术细节更详细地探讨 DEITYBOUNCE 恶意软件的架构。这些细节提供了很多超出我们想象的有价值的线索。但在这之前，我们需要了解一些重要的背景知识。

戴尔 Power-Edge 硬件分析

我们首先介绍第一项背景知识。在上一章中，我们了解到 DEITYBOUNCE 针对戴尔 PowerEdge 1850/2850/1950/2950 RAID 系列服务器。因此，我们需要进一步分析这些服务器，以便更好地了解 DEITYBOUNCE 的执行环境。大家可以从以下链接下载相关服务器的规范：

1.戴尔 PowerEdge1950 规范：
http://www.dell.com/downloads/global/products/pedge/en/1950_specs.pdf

2.戴尔 PowerEdge2950 规范：
http://www.dell.com/downloads/global/products/pedge/en/2950_specs.pdf

服务器规范是相当普通的。但是，如果仔细看一下这两种服务器的存储选项，你会发现它们都具有使用 RAID 控制器的选项。RAID 控制器的类型包括 PERC5/i、PERC4e/DC 或 PERC5/e。DEITYBOUNCE 技术细节提到 RAID 的存在是其中一个硬件“要求”，所以我们将重点分析 RAID 控制器硬件。

现在，我们更详细地分析。您也可以从以下链接下载戴尔 PERC（PowerEdge 可扩展 RAID 控制器）系列 RAID 控制器的用户指南：
ftp://ftp.dell.com/Manuals/Common/dell-perc-4-dc_User's%20Guide_en-us.pdf。尽管该文件只是一个用户指南，但是它提供了重要的信息，如下所示：

1. PERC 代表 PowerEdge Expandable RAID Controller（PowerEdge 可扩展 RAID 控制器）。这意味着 PERC 系列 RAID 控制器或者是戴尔“白色标签”的，或者是由戴尔内部开发的。

2.有几种类型的 PERC RAID 控制器。带有 XX/i 标识的是集成版本， XX/SC 意味着

RAID 控制器是单信道，XX/DC 意味着 RAID 控制器是双信道，XXe/XX 表示 RAID 控制器使用 PCIe（PCI Express，外设部件互连总线扩展接口）而非 PCI（外设部件互连标准）总线。如果最后一个标识缺少，则意味着 RAID 控制器使用 PCI 而非 PCIe。

3.所有 PERC 变种都有 1MB 的板载闪存 ROM。需要注意的是，这不是 PowerEdge 服务器主板闪存 ROM，而是 PERC RAID 控制器（独家）闪存 ROM。它用于初始化和配置 RAID 控制器。

4.所有 PERC 变种都有 NVRAM（非易失性随机访问存储器）来存储其配置数据。NVRAM 位于 PERC 适配器主板，除非 PERC 被集成到主板上。

从固件代码的角度来看，PERC RAID 控制器闪存 ROM（1MB）是很大的。因此，任何人都可以插入一个先进的恶意软件固件级模块。

我没有找到戴尔 PowerEdge1850/2850/1950/2950 BIOS 芯片尺寸的信息。然而，它们的压缩 BIOS 文件大于 570KB。因此，认为其主板 BIOS 芯片尺寸至少为 1MB 是合理的，因为据我所知，没有任何闪存 ROM 芯片（用于存储 BIOS 代码）的大小界于 570KB 和 1MB 之间。最接近 570KB 的闪存 ROM 大小为 1MB。这一事实也表明除了 RAID 控制器扩展 ROM，BIOS 恶意软件也有很大的机会植入主板 BIOS。

IPL（初始程序加载器）设备入门

我们需要了解的第二项背景知识涉及到 IPL 设备。根据 BIOS 引导规范，RAID 控制器或其他存储控制器吸引固件恶意软件的原因就在于它们是 IPL 设备。BIOS 引导规范和 PCI 规范决定了：如果 IPL 设备正在使用中，则 IPL 设备固件必须在启动时执行。IPL 设备固件大多数作为 PCI 扩展 ROM 执行。因此，假设 IPL 设备正在使用中，则 IPL 设备总是执行。这一事实导致固件级恶意软件可以驻留在 IPL 设备固件中，尤其是当该恶意软件需要在每次启动时执行或在启动的某些触发条件下执行。

有关 IPL 设备固件执行的详细信息，请参阅：

https://sites.google.com/site/pinczakko/low-cost-embedded-x86-teaching-tool-2。建议您仔细阅读该文章中的 PCI 扩展 ROM 的 BCV（引导连接向量）部分。引导期间，系统 BIOS 调用/跳转到 BCV，以便启动引导加载程序，然后引导加载程序加载并执行该操作系统。BCV 在存储控制设备的 PCI 扩展 ROM 中执行。因此，戴尔 PowerEdge 服务器中的 PERC RAID 控制

器应该执行 BCV 并符合 BIOS 引导规范。

IPL 设备 PCI 扩展 ROM 也有特殊性。在一些 BIOS 实现中，它总是被执行，不管 IPL 设备是否正被使用。其原因是，BIOS 代码很可能只检查其 PCI 配置寄存器中的 PCI 设备子类代码和接口代码。请参阅下述文章中的“PCI 即插即用扩展 ROM 特殊性”部分：https://sites.google.com/site/pinczakko/low-cost-embedded-x86-teaching-tool-2#pci_pnp_rom_peculiarity。

SMM（系统管理模式）入门

需要了解的第三项背景知识是系统管理模式。SMM 恶意软件的开创性运作可以在 Phrack 电子杂志文章《真正的 SMM Rootkit :逆向和钩挂 BIOS SMI 处理程序》一文中找到，链接为： http://www.phrack.com/issues.html?issue=66&id=11#article。在该背景下，SMI 是指系统管理中断。上述文章包含了解 SMM rootki 如何运作所需的知识。

不过，上述文章中有个问题需要更新。最近和现在的 CPU 不再使用 HSEG（高段内存）来保存 SMM 代码了，只有 TSEG（顶段内存）用于这一目的。如果您不熟悉 HSEG 和 TSEG，可以参阅下述两个文章来了解 CPU 内存中的 HSEG 和 TSEG 位置：http://resources.infosecinstitute.com/system-address-map-initialization-in-x86x64-architecture-part-1-pci-based-systems/；http://resources.infosecinstitute.com/system-address-map-initialization-x86x64-architecture-part-2-pci-express-based-systems/。这意味着，考虑到最大向前兼容性，DEITYBOUNCE 很可能仅在其 SMM 组件中使用 TSEG。

通过软件 SMI（系统管理中断）进入 x86/x64 中的 SMM 是相当简单的。您所要做的是向特定的 I/O 端口地址写入一个特定值。芯片组会将写入操作理解为进入 SMM 中的请求，因此该芯片组向 CPU 发送一个 SMI 信号，以便进入 SMM。有些 x86/x64 CPU 可以直接拦截这样的 I/O 写入操作，并直接把它解释为进入 SMM，而无需通过向芯片组发送任何请求。进入 SMM 的完整算法如下所示：

1.通过 SMM“激活”端口初始化 DX 寄存器（数据寄存器）。通常，SMM“激活”端口是端口 0xB2。然而，也可以是其他端口，这取决于具体的 CPU 和芯片组的组合。关于详细信息，您可以参考其数据表。

2.用 SMI 命令值初始化 AX 寄存器（累加寄存器）。

3.通过向 DX 寄存器的输出端口写入 AX 值来进入 SMM。

至于将参数传递给 SMI 处理程序例程的方法，这篇 Phrack 文章并未涵盖。因此，我们需要分析一下。

有些 SMM 代码（SMI 处理程序）需要与其他 BIOS 模块或操作系统中运行的软件进行通信。通信机制是通过 SMM 代码和 SMM 之外运行的代码之间的参数传递实现的。总的来说，BIOS 模块和 SMI 处理程序之间的参数传递使用以下机制之一：

1.通过 GNVS（全局非易失性存储器）。GNVS 是 ACPI（高级配置和电源管理接口）规范 4.0 版的一部分，被命名为 ACPI NVS（ACPI 非易失性存储器）。然而，在一些专利说明书中，NVS 代表非易失性休眠内存，因为 NVS 在 RAM 中占据的内存区域存储了一些数据，即使系统处于睡眠模式下也可保存。两个术语指得是 RAM 中的同一区域。因此，命名的差异可以忽略。GNVS 或 ACPI NVS 是 ACPI 子系统管理的 RAM 的一部分。它存储各种数据。GNVS 不由操作系统管理，但是可以通过 ASL（ACPI 源语言）接口访问。在 Windows 中，该区域的各部分可以通过 WMI（Windows 管理规范）接口访问。

2.通过 x86/x64 体系结构的 GPR（通用寄存器），也就是，RAX/ EAX，RBX/ EBX，RDX/ EDX 等。在该技术中，一个物理地址指针经由 GPR 传递到 SMI 处理程序代码。由于系统状态（包括寄存器值）被保存到“SMM 保存状态”，SMM 区域（SMI 处理程序）的代码能够读取指针值。美中不足的是：SMI 处理程序和传递参数的代码必须满足“调用协议”，即使用哪个寄存器。

了解 BIOS 模块和 SMI 处理程序之间的参数传递是非常重要的，因为 DEITYBOUNCE 在运行时广泛地使用这种机制。我们将在后面的章节中更详细地介绍。

DEITYBOUNCE 体系结构的初期形式

与其它软件一样，我们可以通过执行环境来推断 DEITYBOUNCE 恶意软件的体系结构。在简介部分，NSA ANT 服务器文档中提到了三个技术提示。我们将逐一进行深入研究，以了解 DEITYBOUNCE 的可能架构。

需要 RAID 硬件意味着 DEITYBOUNCE 包含植入到 RAID 控制器 PCI 扩展 ROM 中的恶意软件。用于戴尔 PowerEdge 1950 服务器的 RAID 控制器是 PERC5/i，PERC4e/DC 或 PERC5/E 适配卡。所有这些 RAID 控制器都是 PCI 或 PCIe RAID 控制器。必须要注意的是，

PCI 扩展 ROM 还包括 PCIe 扩展 ROM，因为这两者几乎是相同的。我在另一篇文章中介绍了 PCI 扩展 ROM 恶意软件的基础信息。请参阅 http://resources.infosecinstitute.com/pci-expansion-rom/。

DEITYBOUNCE 投放的有效载荷意味着 DEITYBOUNCE 基本上是一个“第二阶段”恶意软件投放器--第一阶段是 ARKSTREAM 恶意软件投放器。DEITYBOUNCE 大概只能提供以下两个核心功能：

1.第一个功能是作为持续性和隐蔽的恶意软件模块投放器。

2.第二个功能是向操作系统特定的恶意软件模块提供“隐蔽的”控制功能。该恶意软件模块可以在操作系统初始化时运行，或者在操作系统运行时运行，或者在两种情况下运行。操作系统特定的恶意软件通过 SMM 调用与 DEITYBOUNCE 通信。

上述 DEITYBOUNCE 核心功能表明 DEITYBOUNCE 运行可能需要三种恶意软件组件，如下所示：

1.持续性的“被感染的”PCI 扩展 ROM。该模块包含一个配置 DEITYBOUNCE 执行频率的例程。该例程可能将配置存储在 RAID 控制器 NVRAM(非易失性随机访问存储器)中。该模块还包含了被污染的中断 13h 处理程序，这些处理程序可以通过 SMI 调用其他程序，来修复当前加载的操作系统的内核。

2. 植入到 PowerEdge 主板 BIOS 的 SMI 处理程序代码 ,这些代码服务于从“被感染的”RAID 控制器 PCI 扩展 ROM 的软件 SMI 调用。

3.运行于 Windows 2000、Windows Server2003 或 Windows XP 的特定操作系统的恶意软件有效载荷。

到此，我们已经了解了 DEITYBOUNCE 恶意软件组件。这并不意味着我们能够了解该恶意软件的确切结构，因为它有多个运行组件的途径。不过，我在本文中介绍了目前最有可能的架构。这是一个合理的猜测，但是由于我没有样本 DEITYBOUNCE 二进制文件来支撑该结论，可能会有一些不准确的地方。考虑到 x86/x64 固件架构的本质，我觉得这一猜测应该是足够接近事实的。如果您能够提供 DEITYBOUNCE 二进制样本，那么我非常乐意进行分析。

DEITYBOUNCE 体系结构

在确定 DEITYBOUNCE 的架构之前，我们需要做出几个假设。这样能够更容易地了解 DEITYBOUNCE 的技术细节。以下是我的假设：

1.戴尔 PowerEdge 目标所用的 BIOS 是一个传统的 BIOS，而非 EFI/UEFI（可扩展固件接/统一的可扩展固件接口）。这个假设符合 NSA ANT 服务器文档的描述，其中提到目标操作系统是 Windows2000/XP/2003。所有这些操作系统都没有提供成熟的 EFI/UEFI 支持。所有的用户手册，其中包括 BIOS 设置用户手册，也没有显示出与 UEFI/EFI 支持有关的任何菜单，例如在传统 BIOS 模式下启动。因此，假设它是传统的 BIOS 是合理的。此外，在 DEITYBOUNCE 发动攻击时，市场上的 EFI/UEFI 支持也还很不成熟。

2.引导过程中，用于修复操作系统内核的自定义 SMI 处理程序例程大于主板 BIOS 中的可用空间。因此，该例程必须被放置到两个独立的闪存 ROM 存储器，即 PERC RAID 控制器 ROM 闪存芯片和戴尔 PowerEdge 主板 ROM 闪存芯片。事实可能并非如此，但我们只是做一个假设，因为即使基本的 NTFS（新技术文件系统）驱动器在压缩时也至少需要几万字节的空间，更不用说旨在修复三种不同操作系统的内核的复杂恶意软件了。

以上假设对我们的 DEITYBOUNCE 架构分析影响颇大。第一个假设是 DEITYBOUNCE 的执行分为两个阶段。第二个是，引导时（中断 19h）修复操作系统内核的恶意软件代码在 SMM 中运行。

现在，我们来分析 DEITYBOUNCE 的执行阶段，如下所示：

1. 阶段 1：POST（加电自检）时的 PCI 扩展 ROM。在此阶段，DEITYBOUNCE 在主板的 SMRAM（系统管理 RAM）区域安装其他恶意 SMI 处理程序。我们的假设是，主板 BIOS 还未锁定 SMRAM 区域，因此 SMRAM 范围仍是可写入的。SMRAM 是系统内存（RAM）中的一个范围，专门用于 SMM 代码和数据。SMRAM 的内容只能在锁定后通过 SMI 访问。在大多数英特尔北桥芯片组或最近的 CPU 中，SMRAM 是由控制 TSEG（高段内存）区域的寄存器控制的。通常，在英特尔芯片组的文档中，寄存器被称为 TSEG_BASE。

2. 阶段 2：引导时中断 13h 执行。在 PERC RAID 控制器中，PERC RAID 控制器 PCI 扩展 ROM 中的中断 13h 处理程序被用恶意代码“修复”了，以服务于中断 19h 调用（引导）。中断 19h 通过调用中断 13h 的函数 02h，将引导加载程序复制到 RAM，从驱动器中读取扇

区，并跳转到扇区。DEITYBOUNCE 并不感染引导加载程序。然而，DEITYBOUNCE 感染了中断 13h 处理程序。该“修复的”中断 13h 处理程序将会修改 RAM 中加载的操作系统内核。

图 2 展示了 DEITYBOUNCE 的执行阶段 1， 图 3 则展示了 DEITYBOUNCE 的执行阶段 2。

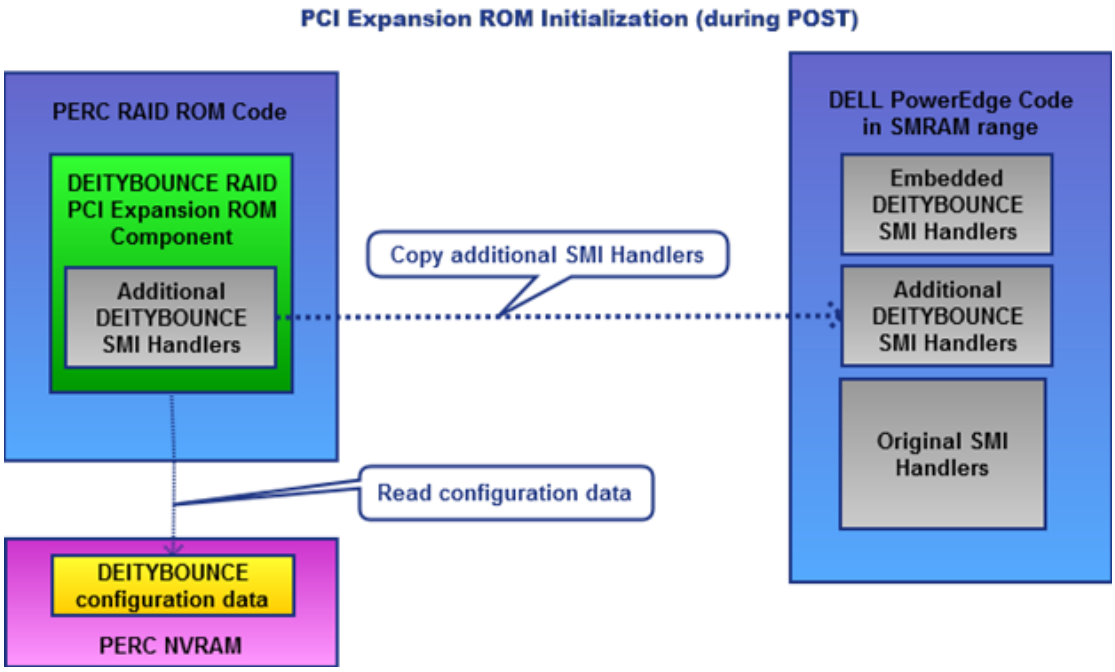

图 2：DEITYBOUNCE 执行阶段 1

如图 2 所示，DEITYBOUNCE 阶段 1 执行发生在加电自检的 PCI 扩展 ROM 初始化阶段。如果您不熟悉 BIOS 初始化 x86/x64 系统（又名加电自检）的详细步骤，请参阅我的《x86/x64 系统地址映射初始化》一文，链接是：http://resources.infosecinstitute.com/system-address-map-initialization-in-x86x64-architecture-part-1-pci-based-systems/。

我们知道，PCI 扩展 ROM 初始化是由主板 BIOS 发起的，请参阅《PCI 扩展 ROM 中的恶意代码执行》一文（http://resources.infosecinstitute.com/pci-expansion-rom/）。主板 BIOS 调用 INIT 函数（初始化函数，PCI 扩展 ROM 起始处的偏移 03h 处），通过 PCI 扩展 ROM 启动附加板初始化。这就是 DEITYBOUNCE 执行的第一阶段。在该情况下，附加板是 PERC PCI/PCIe 板或集成在 PowerEdge 主板上的 PERC 芯片。

图 2 展示了以下执行步骤：

1. PERC RAID PCI 扩展 ROM 从 INIT 入口点执行。

2.被感染的 ROM 代码从 RAID 控制器 NVRAM（非易失性随机访问存储器）读取 DEITYBOUNCE 配置数据。

3.被感染的 ROM 代码将 DEITYBOUNCE 的额外 SMI 处理程序复制到主板主内存（系统 RAM）的 SMRAM（系统管理 RAM）区域。

4.被感染的 ROM 代码根据需要修复 SMRAM 内容的校验和。

一旦上述步骤完成后，SMRAM 就包含所有的 DEITYBOUNCE 的 SMI 处理程序了。图 2 显示了 SMRAM 包含的一些“嵌入”DEITYBOUNCE SMI 处理程序早就存在于 SMRAM 中。嵌入的 DEITYBOUNCE 组件被注入到主板 BIOS。这就是为什么它早已经存在于 SMRAM。

图 2 展示了存储在 PERC 附加板 NVRAM 的 DEITYBOUNCE 配置数据。这是一种巧妙和隐蔽的存储配置数据的方法。有多少反恶意软件工具扫描附加板 NVRAM？我从来没有听说过。

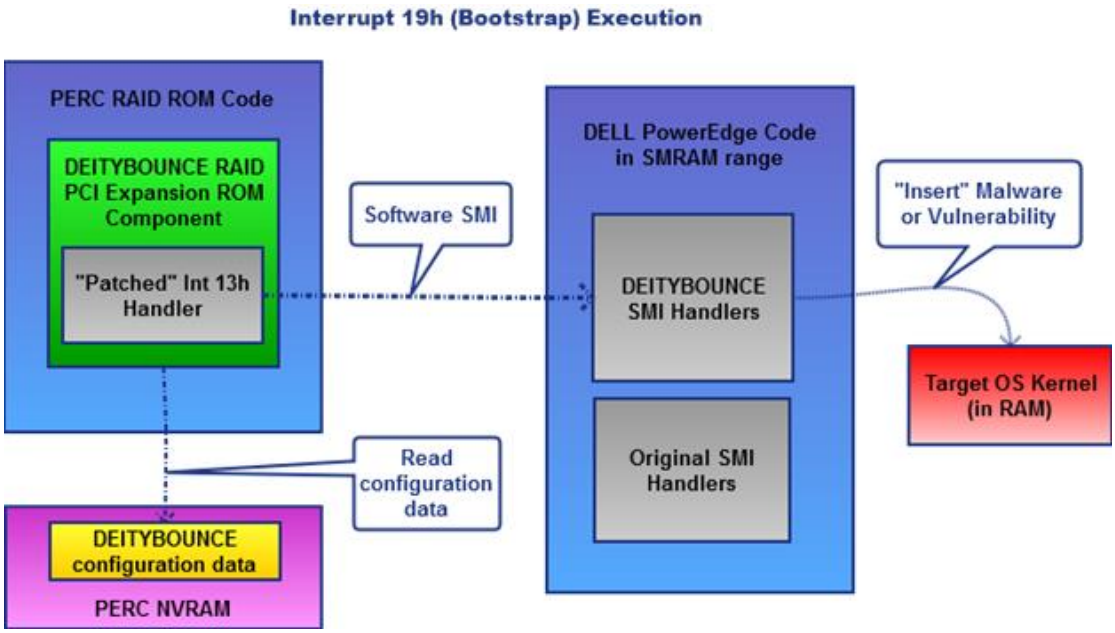

图 3：DEITYBOUNCE 执行阶段 2

现在，我们来分析 DEITYBOUNCE 执行的第二阶段。这一阶段包括若干步骤，如下所示：

1.主板 BIOS 通过中断 19h（引导）执行 PERC RAID 控制器 PCI 扩展 ROM 例程。此时

的 PCI 扩展 ROM 的入口点是它的 BCV（引导连接向量），而不是 INIT 函数。中断 19h 调用中断 13h 函数 02h 以便将启动设备（在这种情况下是指 RAID 控制器控制的硬盘驱动器）的第一个扇区读取到 RAM，然后跳转到这一扇区，并启动引导加载程序。

2.被感染的 PCI 扩展 ROM 例程包含自定义中断 13h 处理程序。当引导加载程序和部分操作系统初始化代码调用中断 13h 时，这个自定义的中断处理程序就会执行。自定义例程包含原始中断 13h 处理程序逻辑。但是它添加了感染正在加载的操作系统的内核的例程。中断 13h 提供服务来读取/写入/查询存储设备。因此，恶意编码器可以修改中断 13h 处理程序例程的内容，以篡改被加载到 RAM 中的内容。图 3 表明自定义中断 13h 包含一个例程，该例程可以通过软件 SMI 调用 DEITYBOUNCE SMI 处理程序。DEITYBOUNCE SMI 处理程序又包含一个例程，用于安装恶意软件或在操作系统仍然处于初始化阶段时激活目标操作系统内核的某些漏洞。自定义中断 13h 处理程序的执行取决于 DEITYBOUNCE 配置数据。图 3 所示的自定义中断 13h 处理程序的相关 DEITYBOUNCE 配置数据存储于 PERC RAID 控制器 NVRAM。

DEITYBOUNCE 阶段 2 的步骤 1 和 2 执行后，目标操作系统就会包含一个漏洞或者恶意软件。请注意，该漏洞或恶意软件只存在于 RAM，因为操作系统被 DEITYBOUNCE 修改的情况只发生于 RAM 中，而不是在永久存储设备（硬盘驱动器或固态硬盘）。

至此，我们已经比较详细地了解了 DEITYBOUNCE 可能的运作方法。但是，这只是我对于 DEITYBOUNCE 初步评估的结果。因此，必然有不准确之处。

结论：DEITYBOUNCE 技术目的的假设

与“普通”恶意软件相比，DEITYBOUNCE 具备两个不可否认的战略价值：

1. DEITYBOUNCE 提供了一个隐蔽的方式来改变所加载的操作系统，而且不会在存储设备（即硬盘驱动器或固态硬盘）上留下痕迹，以避免通过“普通”计算机取证程序被发现。为什么呢？因为当操作系统被加载到 RAM 时会被操纵，存储设备中的操作系统安装则不会被感染。SMM 代码执行提供了一种隐蔽的代码执行方法，即另一方扫描器对操作系统完整性的检查。在这方面，DEITYBOUNCE 是一个非常复杂的恶意软件投放器。

2. DEITYBOUNCE 能够持续存在于目标系统，即使操作系统重新安装它也会持续存在。

鉴于 DEITYBOUNCE 提供的功能，可能有一个隐蔽的 Windows rootkit 通过软件 SMI

（系统管理中断）与 DEITYBOUNCE 通信，以便在运行时从 Windows 调用 DEITYBOUNCE SMI 处理程序。目前，我们尚不清楚该 rootkit 的目的，也不清楚 DEITYBOUNCE 是否执行所需的 SMI 处理程序。

作者简介

Darmawan Salihun 自 2002 年以来一直专注于 BIOS 相关的安全研究，他已在《破译者杂志》上出版了 BIOS 逆向工程、BIOS 代码注入技术、BIOS 黑客技术、即插即用 BIOS 和 PCI（外设部件互连标准）协议开发的多篇论文。2006 年，他撰写了《BIOS 反汇编忍术》一书，介绍了他对 BIOS 安全研究的结果。他目前仍在从事 BIOS 和 UEFI（统一的可扩展固件接口）安全研究。他还是资讯安全研究所的研究员。

27 条评论

1. 2014 年 1 月 30 日，Rommelfs：计算散列或还未发生。

2. 2014 年 3 月 3 日，Anon：Rootkit 隐藏自己，你不能在不采用连接的芯片读卡器实际提取固件的情况下“计算散列”（假设它们还没有对防御这种方法，但它们很可能已经防御了）。

3. 2014 年 8 月 17 日，Menachem wuz hear：BIOS？拆下 IC（可能是串行闪存），把它连接到一个阅读器或 Arduino，并提取出代码。发布代码，并指出它与 OEM 的不同之处。

或者这一切都只是科幻小说。

4. 2014 年 1 月 30 日，Tom：感谢您分享这篇精彩的文章，文章内容清晰而且证据充分。我把文章链接贴在了布鲁斯·施耐德的网站上（他提供了一个讨论 NSA ANT 目录的论坛）。https://www.schneier.com/blog/archives/2014/01/for DeityBounce。

正如你在文章第一段中所说的，GodSurge 类似于 DeityBounce。它也针对 Dell PowerEdge 服务器，但要求大型的 FluxBabbitt 硬件植入物并利用 JTAG 调试接口。最好的 JTAG（联合测试工作组）似乎是在：http://www.alexforencich.com/wiki/en/reverse-engineering/jtag/start。

ANT 目录中其他提到 BIOS 的项目是 IRONCHEF、SWAP、STUCCOMONTANA、SCHOOLMONTANA、SIERRAMONTANA、SOUFFLETROUGH，可能还有 GOURMETTROUGH 和 FEEDTROUGH。

希望您能研究一下其中一些项目，并撰写出此等深度的文章。

5. 2014 年 1 月 30 日，Darmawan Salihun：Tom 你好。本文仅仅是系列文章的第一部分。我会尽快分析其他项目。据我所知，GodSurge 是一个完全不同类型的攻击。至少还有一个与它相当类似。但是，我不会剧透啦，因为它是系列文章中的下一个。

6. 2014 年 1 月 31 日，Tom：我期待着您的下一篇文章，您所提供的背景知识能够帮助信息匮乏的人加快研究速度。所以，再次感谢您的慷慨分享，这是互联网的最大优点了。

如果这些东西出现于真实的目标计算机中，您肯定是“去看看”并进行分析的那类人。随着 TCP 端口 32764 事件的发生 ,在线工具很快出现了 ,让人们检查其端口是否是开放的。现在，我们有类似 ZMAP 的开源工具，可以在一小时内扫描整个互联网的某些属性。大胆地想象一下，我们能否找到在全球范围内消除这些植入物的方法？

以下链接是使用 JTAG（联合测试工作组）在硬盘驱动器上安装恶意代码的教程。

http://spritesmods.com/?art=hddhack&page=3

7. 2014 年 1 月 31 日，Matthew Beddoes：谢谢您分享本文，现在我明白它是如何运作的了，可以尝试找出解决方法。

8. 2014 年 2 月 1 日，Darmawan Salihun：Tom，谢谢你分享链接。其实，大约在 2013 年中期，我准备研究 HDD（硬盘驱动器）固件，但是由于其他紧要工作而终止了。还有其他的方式来获得 HDD 固件，前提是它不能被损坏。不管怎么说，这篇 BIOS 文章是我周末研究的结果。

9. 2014 年 2 月 3 日，Darmawan Salihun：文章没有涉及的一个话题是“远程自擦除”，该功能可以远程删除恶意软件，以防它变得拙劣起来。在下一篇文章中，我将会分析这一问题。

10. 2014 年 2 月 3 日，Darmawan Salihun：“远程自擦除”是军用电子中的一个标准机制，旨在防止敌方了解自己的技术。

11. 2014 年 3 月 3 日，Anon： 所有惠普产品也有一个非常容易利用的后门，其 iLo 固件在 IPMI（智能平台管理接口）端口泄漏了 iLo 管理员凭据，使得任何人都能够远程修改服务器固件。

12. 2014 年 3 月 24 日，Vitor：Darmawan Salihun 您好，我正在开发一套 Linux 软件包，使用 SMI（系统管理中断）来查询设备的状态。内核模块的主要功能如下所示。我想请您帮我看看代码究竟是如何运行的。

长话短说，命令行“out %%al,$0xb2\n\t”是为了调用 SMI，但我不太明白之后的命令行“out %%al,$0×84\n\t”。您可否帮帮我呢？开发网站是“https://launchpad.net/i8kutils”。

```
static int i8k_smm(struct smm_regs *regs)
{
ktime_t calltime, delta, rettime;
unsigned long long duration;
int ret;

calltime = ktime_get();

int rc;
int eax = regs->eax;

#if defined(CONFIG_X86_64)
asm volatile("pushq %%rax\n\t"
  "movl 0(%%rax),%%edx\n\t"
  "pushq %%rdx\n\t"
  "movl 4(%%rax),%%ebx\n\t"
  "movl 8(%%rax),%%ecx\n\t"
  "movl 12(%%rax),%%edx\n\t"
  "movl 16(%%rax),%%esi\n\t"
  "movl 20(%%rax),%%edi\n\t"
  "popq %%rax\n\t"
  "out %%al,$0xb2\n\t"
  "out %%al,$0×84\n\t"
  "xchgq %%rax,(%%rsp)\n\t"
  "movl %%ebx,4(%%rax)\n\t"
  "movl %%ecx,8(%%rax)\n\t"
  "movl %%edx,12(%%rax)\n\t"
  "movl %%esi,16(%%rax)\n\t"
  "movl %%edi,20(%%rax)\n\t"
  "popq %%rdx\n\t"
  "movl %%edx,0(%%rax)\n\t"
  "pushfq\n\t"
  "popq %%rax\n\t"
  "andl $1,%%eax\n"
:"=a"(rc)
: "a"(regs)
: "%ebx", "%ecx", "%edx", "%esi", "%edi", "memory");
#else
asm volatile("pushl %%eax\n\t"
  "movl 0(%%eax),%%edx\n\t"
  "push %%edx\n\t"
```

```
"movl 4(%%eax),%%ebx\n\t"
"movl 8(%%eax),%%ecx\n\t"
"movl 12(%%eax),%%edx\n\t"
"movl 16(%%eax),%%esi\n\t"
"movl 20(%%eax),%%edi\n\t"
"popl %%eax\n\t"
"out %%al,$0xb2\n\t"
"out %%al,$0×84\n\t"
"xchgl %%eax,(%%esp)\n\t"
"movl %%ebx,4(%%eax)\n\t"
"movl %%ecx,8(%%eax)\n\t"
"movl %%edx,12(%%eax)\n\t"
"movl %%esi,16(%%eax)\n\t"
"movl %%edi,20(%%eax)\n\t"
"popl %%edx\n\t"
"movl %%edx,0(%%eax)\n\t"
"lahf\n\t"
"shrl $8,%%eax\n\t"
"andl $1,%%eax\n"
:"=a"(rc)
: "a"(regs)
: "%ebx", "%ecx", "%edx", "%esi", "%edi", "memory");
#endif

rettime = ktime_get();
delta = ktime_sub(rettime, calltime);
duration = (unsigned long long) ktime_to_ns(delta) >> 10;
printk(KERN_DEBUG "i8k_smm function took %lld usecs\n", duration);

dump_stack();

if (rc != 0 || (regs->eax & 0xffff) == 0xffff || regs->eax == eax)
return -EINVAL;

return 0;
}
```

13. 2014 年 3 月 27 日，Darmawan Salihun：@anon：BIOS/UEFI（统一的可扩展固件接口）安全研究人员研究 IPMI（智能平台管理接口）已经有一段时间了。

@vitor： 我不是很确定端口 0×84 的用途。这也可能是系统特定的。我也还没有读过 AMD 的 BKDG（《BIOS 与内核开发指导》）一文。也许，文章中会有所答案。

14. 2014 年 4 月 11 日，Peter：这个恶意软件能否感染那些正在使用 SINIT/SENTER 进行初

始化的操作系统？

15. 2014 年 4 月 20 日，Darmawan Salihun：@peter：我不确定该恶意软件有效载荷能否感染这种操作系统。但是，如果碰巧 NSA 通过 SHA-1 校验和的某种“后门”破解了哈希保护，那么我们就没什么好运了。

16. 2014 年 6 月 28 日，Jason： 我已经遭受持续性攻击 15 个月了，您的文章非常有帮助。如果您仍然需要 BIOS 样本（还有 PCIe 扩展 ROM 样本），请务必告知。

1.如果您能告诉我怎么从硬盘中读取固件，那就帮我大忙了。

2.如何实现针对 IPMI（智能平台管理接口）的最好保护方法？从 Windows 中完全将它删除并删除 CMD 文件么？还是？

祝好。

17. 2014 年 7 月 8 日，BadBIOSvictim：Jason 提出能够提供 BIOS 和 PCIe 扩展 ROM 样本。如果能有一个简单的指导，我也想提供样本。否则，我就把我已弃用的东芝笔记本电脑提供给任何有兴趣取证和识别植入物的人士。植入物可能类似于 BULLDOZER。我的电子邮件地址是 bluelotus@openmailbox.org。谢谢。

18. 2014 年 8 月 15 日，No：ANT 目录不是斯诺登泄露的，可能是另一个泄密者。

19. 2014 年 8 月 16 日，Dave Gilbert：这篇文章似乎认为所有被注入的代码通过中断 13 或 SMI（系统管理中断）运行于 x86 系统，但是由于 PERC（PowerEdge 可扩展 RAID 控制器）有一个可编程的 RAID 控制器（我认为是 ARM 或其他重要的控制器），那么代码就可能运行在 PERC 吗？然后，它会通过启动在主机内存中注入块，而不显示任何磁盘变化。而事实上，它可能以定义良好的引导顺序（而不是在磁盘检查时）返回“特别”版本的块。

尤其是在旧的 PCI（外设部件互连标准）系统中，我认为 RAID 控制器可以直接内存读取有意思的 RAM 片段，而不需要涉及 x86 系统。

20. 2014 年 8 月 17 日， buckwii： 我很肯定至少我有一台机器被感染了。它采用戴尔 A02 BIOS。可否告诉我应该提供给您什么日志，以便您提出意见？我只运行 Linux 发行版。我安装了 Mint 15，还使用了 Debian 磁盘和用 Parted Magic 磁盘。谢谢。我的电子邮箱地址：ham.hamlin@aol.com。

21. 2014 年 8 月 17 日，Paolo：PCI 选项 ROM 不应该被允许向 SMRAM 写入任意内容（其实，我认为在超级复杂的 UEFI SMM 架构中，这是被禁止的）。所以 DEITYOUNCE 至少在一定程度上利用了戴尔 SMM（系统管理模式）代码漏洞吗？

22. 2014 年 8 月 20 日，Matt：@Paolo，直到英特尔产品出现 UEFI（统一的可扩展固件接口）之后不久，戴尔才转向 UEFI。对于传统系统来说，对写入 SMRAM（系统管理 RAM）的内容没有什么规范。BIOS 可能在创建时生成了一个图像，然后加载图像并关闭通道。其他的可能具备向 SMRAM 添加处理程序的模块，这有点像 UEFI PI 规范 IIRC 的规定。戴尔的 BIOS 可能拥有允许它这样做的模块化方法。该漏洞利用代码需要知道加载处理程序的函数使用的“调用协议”，而且能够加载这些协议。请记住，这都是 x86 系统的 16 位汇编，所以真的没有任何标准。NSA 不得不投大量时间，对 BIOS 和 RAID 控制器固件进行逆向工程。这就是为什么对这个恶意软件来说，硬件是这么具体。但是，很多携带大量流量的相同服务器的数据中心也值得一试。

23. 2014 年 8 月 17 日，Smiley：感谢您分享这篇文章。文章中提及了我们使用的戴尔机器。这就是为何我们之后再也不会为公司和员工购买戴尔的任何东西。

24. 2014 年 8 月 17 日， Michael Wales： SneakerNet 并不是一个项目术语（因此不采用大写），它只是一个俚语，代表数据通过物理手段移动。更有趣的是，该术语常用来指不同类型的网络之间的数据移动。

例如，“这个文档是在 SIPR（机密 IP 协议路由网）上做的，但是没有什么内容被列为 SECRET（机密），我需要用电子邮件发送。我应该将它 SneakerNet 到 NIPR（非机密 IP 协议路由网）。“

25. 2014 年 8 月 20 日，Matt：这些戴尔服务器的 BIOS 是几乎都是传统的。它们花了一段时间切换到 UEFI（统一的可扩展固件接口）。

1MB 闪存部分的假设可能是正确的。当今，BIOS 通常使用较大的闪存，但是它们使用英特尔平台上的芯片组功能（称为描述符模式）来共享多个组件的部分（例如局域网控制器）。

UEFI 安全启动可能会停止 PCI（外设部件互连标准）设备仍然有选项 ROM，但它们是遵循 UEFI 驱动程序模式的可执行图像。随着安全启动的启用，它们必须被签名。尽管如此，了解签名过程中漏洞也是很有趣的。

这同样适用于操作系统使用的引导加载程序，操作系统将其用于 IPL（信息程序装载）。它还没有被广泛使用，而且公众一直不信任它（据说，受到收买的代工工厂理论上可以利用该系统锁定特定的操作系统，但这个问题现在应该已经解决了）。

直到现在，至少对基于英特尔的系统来说，切换到 UEFI 应该已经完成了。对于 AMD 平台，我倒不是很确定。现在为止发现过任何针对 UEFI 的攻击吗？

26. 2014 年 9 月 19 日，IRATEMONKEY：我有一台台式计算机，将其称为 PC-A，我曾经三次将硬盘驱动器格式化（使用 killdisk 至少执行一次“一通零”），以便安装 Windows7 64 位系统。

格式化计算机并重新安装 Windows 系统后，我做的第一件事是将安装 TrueCrypt 7.1a，然后在执行任何其他操作之前进行充分的磁盘加密（包括安装任何设备驱动程序或将其连接到网络之前等等）。

使用过 TrueCrypt 全磁盘加密的人都知道，它会要求你在使用 FDE（全磁盘加密）之前刻录应急光盘。我不想每次都浪费光盘，所以我总是下载和使用 WinCDEmu，以便绕过这个刻录要求。

然而，最近，当我重新格式化，擦除内容并在 PC-A 上重新安装 Windows 后，我注意到，当点击安装并试图安装 WinCDEmu 时，出现了一个奇怪的错误，指出“Microsoft 注册服务器已停止工作”，详细信息显示出一个与 DEP（数据执行保护）有关的“BEX”错误：WinCDEmuContextMenu.dll_unloaded。我在三个不同场合尝试过，每次都是完全从头开始，结果都一样。

我完全没有做过任何硬件改动，没有执行 BIOS 或固件升级。每次我都是使用完全相同的 Windows 7 的 DVD-ROM，通过引导光盘安装操作系统，光盘本身都是没有划痕的。我也一直用完全相同的 WinCDEmu 版本并进行校验，以确保没有软件损坏或文件完整性的问题。我也使用完全相同的 TrueCrypt 版本。事实上，因为我多次这样做了，我知道 TrueCrypt 会要求我刻录应急光盘。安装 Windows 7 后我做的第一件事就是安装 WinCDEmu 3.6，这甚至是在安装 TrueCrypt 之前的。

我甚至尝试在另一台计算机（称为 PC-B，它是气隙的，从未连接到网络）使用同样的 Windows7 64 位 DVD 启动盘，并用完全相同的 WinCDEmu 版本和完全相同的外部 USB 存

储介质，这些都没有任何问题，没有显示错误信息。

所以，我的步骤是完全一样的，什么都没有改变。在疑似感染之前，我至少三次在 PC-A 上使用了完全相同的步骤和软件，从未出现过任何问题或错误。

现在，我还是使用完全相同的步骤，完全相同的硬件，完全相同的软件，但总是显示错误消息。作为一种测试/控制手段，我甚至尝试在两台计算机上使用完全相同的 Windows DVD 安装光盘、完全相同的 WinCDEmu 3.6 版本（校验和），其中一台计算机是气隙的，另一台则不是，它们都没有出现任何问题或错误消息。

在受感染的机器上，当我试图继续执行 FDE（全磁盘加密）并加密主机保护区时，却无法工作。它似乎是工作的，但当我重启计算机做“测试”时，它却无法识别我输入的密码，但我 100％肯定密码是正确的。此外，该硬盘是标准的希捷硬盘，是商用硬件，但是当我将硬盘换到不同的机器上时，它却不能被正确识别，我也无法访问并复制硬盘数据或提取任何数据。看来，只有安装在原来设备上时，它才会工作。

这种情况我从来没有遇到过。因为在重新安装 Windows 之前，我至少执行了一次“一通零”，以擦除整个硬盘的内容，没有任何硬件变化，而且在对新安装的操作系统做任何更改(连接到互联网、更新驱动程序 ,或安装任何其他应用程序等)之前我都会安装 WinCDEmu。这种奇怪现象的唯一解释是我被高级持续性威胁（例如 NSA）攻击了。

截图：

image.bayimg.com/0a91bdcdb399dbf4322f89582af86f3cf70998db.jpg

image.bayimg.com/b42874a45ecde2e24940d33511c7ba81a560e407.jpg

如果情况没有这样巧合，我就不会发现这一点了。

1）当重新格式化计算机时，我有一点强迫症。我经常频繁地（其实没什么必要）格式化计算机。然而，重复运动会产生肌肉记忆。如果你做某件事情 N 次了，但是随后的第 N+1 却发生了变化，那么你就会很容易发现。

2）用相同的 DVD 安装 Windows7 后，我做的第一件事情都是安装 WinCDEmu，然后再安装 TrueCrypt 7.1a。当然在做安装时，我没有执行任何 USB 连接或其他连接，计算机不具备 WiFi 功能，也没有以太网电缆插入。如果我设置了 msconfig/services.msc/gpedit.msc，或安装了其他软件，或更新或安装了固件、BIOS、操作系统和软件驱动程序，那么我可能

会知道为何出现错误消息，但是这些我都没有做过。

3）非常奇怪的事情是：我无法复制硬盘驱动器（即使使用硬件方法），为了进行测试，我把硬盘安装到另一台计算机，看它是否会让我在硬盘上安装 Windows 7 系统。使用可引导的 Windows 7 光盘，它会引导到第一个初始 Windows “能量标志” 画面（就是您选择安装或升级 Windows 的画面出现之前），这个画面会持续 60 秒或以上。这时候，什么都不用做，等着这部分结束就行。至少约 60 秒后，它将切换到安装/升级画面，我选择了重新安装，却弹出消息说磁盘不可引导，Windows 系统无法安装于该盘（即使它已经被 Windows 和 BIOS 识别）。

4）因此，在其中一个测试中，当计算机卡在 Windows 启动画面（在安装/升级选项的画面之前）时，我有些不耐烦，直接切断了计算机的电源。之后，我重新通电，并尝试重新启动/打开计算机，但是它却不执行任何操作，就像是 BIOS 被损坏或 BIOS 闪烁期间被切断电源了。

这怎么可能呢？DVD-ROM 驱动器的固件本身也会被感染吗？如果在受感染的硬盘上有一个固件恶意软件，我将其安装到另一台干净的机器，并尝试从 DVD 安装 Windows7（该 DVD 显然不能被污染），那么恶意软件是如何从硬盘或者硬盘固件跳转到 BIOS 或其他计算机的其他组件的？

27. 2014 年 9 月 19 日，IRATEMONKEY：截图

image.bayimg.com/0a91bdcdb399dbf4322f89582af86f3cf70998db.jpg

image.bayimg.com/b42874a45ecde2e24940d33511c7ba81a560e407.jpg