A Myth or Reality – BIOS-based Hypervisor Threat

Special research prepared by Rubos, Inc. team

(We do independent research on security matters in various domains)

Prepared for DeepSec 2014

Speaker: Mikhail Utin, PhD, CISSP

mikhailutin@hotmail.com

(Questions will be answered after the presentation. Please, submit them to the speaker in writing)

Myths and Reality

Myths and reality intersect and interchange.

Greek myth of Achilles – in modern terms is a “search of an ultimate protection and creation of single point of failure”

Fig.1. *Thetis Dipping the Infant Achilles into the River Styx* (ca. 1625)

Modern Myths

- Ghosts, witches, witchcraft; Ghost hunting have been entertaining but unsuccessful – none apprehended and caged for public exposition ; for believers- reality, for others – a myth
- UFO and UFOlogy – a sort of ghost hunting; believers have a fun and consider a reality, but yet a myth – no alien is available for public demo
- IT Management – some see as reality, but in cases like Case #3 below – just a myth
- Information Security – do we have an official myth or a ghost?
- **A myth about Malicious Hypervisor (Russian Ghost) appeared on Russian hackers' site at the end of 2011. It has all myth's attributes. There were rumors about the post, and the storyteller described it as reality. However, neither he nor somebody else got and caged the ghost for public exhibition**

Russian Ghost Myth – Case #1 and Following Research

We believe that it was real or may still exist, **and we possibly know where it was born and eventually escaped from.**

Our research is about three cases. We are interested to identify not only the "percent of reality", but also how dangerous it is, how to hunt on, and how to protect from:

Case #1 - ***Russian research on Malicious BIOS Loaded Hypervisor*** (approximately 2007 – 2010 years) posted at the end of 2011

Case #2 - ***US Michigan University Virtual-Machine Based Rootkit*** research which was done approximately in 2005 – 2006 years (published 2006)

Case #3 - ***US Michigan University IPMI/BMC (Intelligent Platform Management Interface/Baseboard Management Controller) vulnerability*** research of approximately 2012 – 2013 (published 2013) .

Research Direction

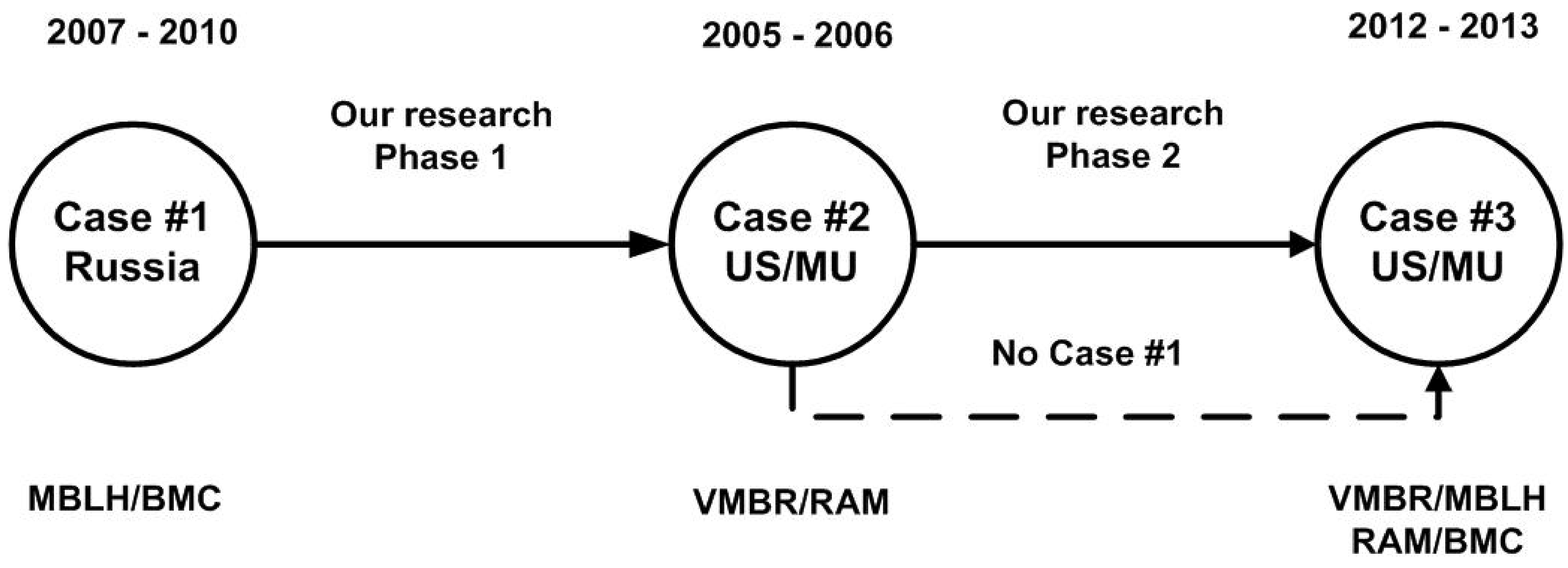


Fig. 2. The process of our research

Case #1 - Russian Ghost – a Real Post and a Myth about Malicious BIOS Loaded Hypervisor (MBLH) (1)

- Was accidently found in the middle of 2012
- **Contains various topics (not technical only) and details**
- Has been published on 12/26/2011
- **Written in Russian and published on Russian hackers' site**

If it were a scam with the intention of misleading, it would be published in English. **The fact that he post is almost unknown relates to its language**.

We translated the text and its summary is published on DeepSec site. Please, see it. Complete text will be available as well.

Reference: Chinese Add-ons: True Stories of virtualization, information security and computer spying; post on http://xakep.ru/articles/58104/ ; 12/26/2011

Translated from Russian, Copyright © DeepSec, GmbH and Rubos, Inc., 2014.

Case #1 – Malicious BIOS Loaded Hypervisor (2)

The project:
Typical Russian computer science project – to develop high performance computer system; no association with information security. **The customer – Kraftway, one of biggest Russian IT companies working closely with Russian government. It supplied Intel motherboards for the project.**
Quote: “I decided to use new Intel processors with the support of hardware virtualization even before the official announcement (in early 2007), and to create a unified computing system … **It required writing compact hypervisor with non-standard functionality. …It is essential to boot the OS on already virtualized platform, so even the first commands of the OS boot loader would be executed in a virtual environment**. Then the hypervisor has to virtualize the CPU real mode and all other operating modes. … it was overstatement to call that as "hypervisor", **I began to use the term "hyper-driver**”.
“ …**thanks to the cooperation with the company Kraftway I had access to pre-production models of processors and motherboards**…”
Comment: pre-production boards were from Kraftway company and labeled “Assembled in Canada”.. They have been used for the design phase.

Case #1 – Malicious BIOS Loaded Hypervisor (3)

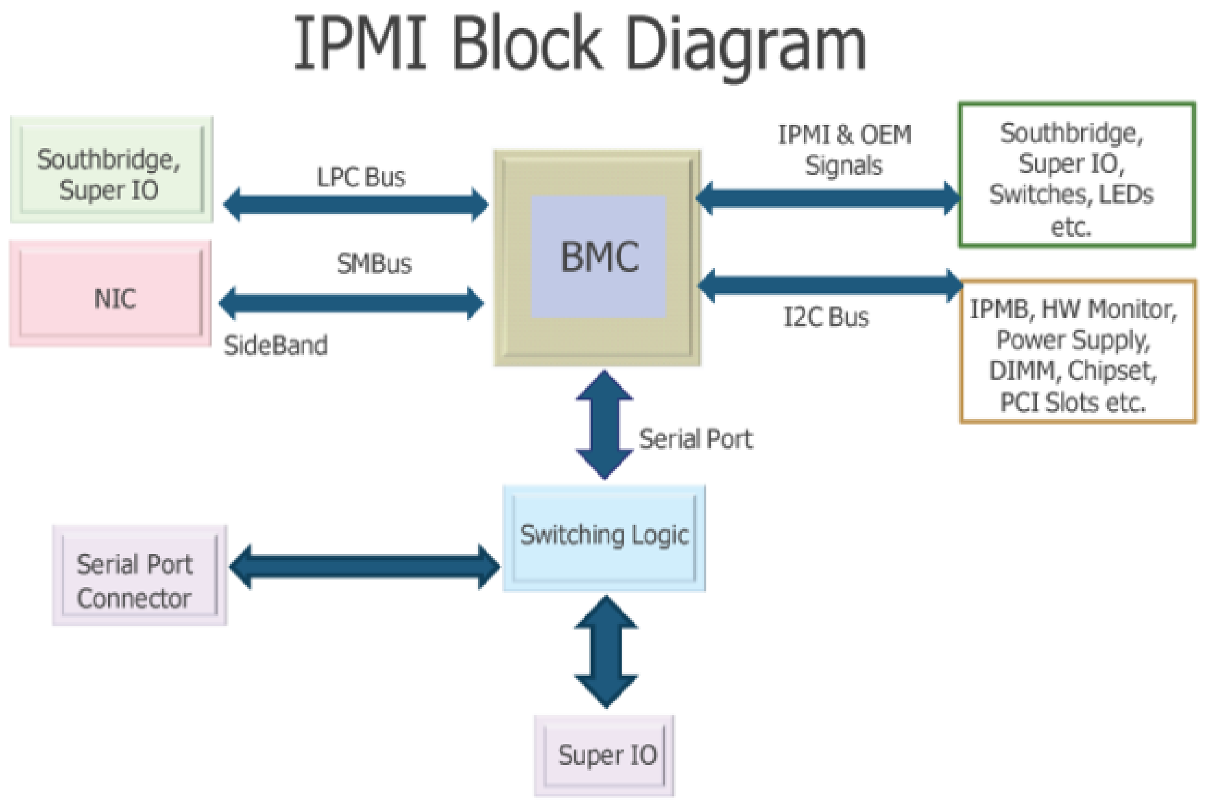
The problem:
"**I had to assemble a system based on first production motherboards**, which had just appeared. **However, the system did not work.** … I began to investigate and finally realized that my system hangs while executing hardware virtualization commands … **and finally get the BIOS code from the official Intel website and reloaded it in the motherboard, and then system started working.**"
The reason:
"**China-made motherboard started to work only when I uploaded the BIOS taken from Intel site. … It became clear that the boards from China contain additional software modules, embedded in the BIOS, and the standard analysis software does not see them**. Apparently, they also worked with hardware virtualization and thus were able to hide the true contents of the BIOS. … **Chinese boards had two software systems working simultaneously on the same hardware virtualization level, which does not allow sharing of resources**." By "two software" he means his "hyper-driver" and embedded malicious hypervisor.

Case #1 – Malicious BIOS Loaded Hypervisor (4)

Comment:
Here is very significant finding **– there is malicious hypervisor embedded in BIOS, and which utilizes hardware virtualization Intel CPU capability. It was possible to identify that because there was a conflict in sharing resources**. The developers of Malicious BIOS-Level Hypervisor (MBLH) did not expect that the system will run the second virtualization software on the same level. **We found official Intel documents that the company indeed has assembly and testing facility (CD1) in Chengdu, China opened in 2005, and another factory opened in Chengdu in 2007 (CD6). However, in Canada we found only Flash Memory group in Vancouver/BC, which is unlikely being involved in assembly process. Thus, there are some doubts concerning the label on the sample boards "Assembled in Canada".**
The home of the MBLH:
"… **it was necessary to begin to figure out in which of the flash chips was a software module that works with hardware virtualization**. … I was not even surprised when found out that **the "add-on" hypervisor starts working just after reloading software in flash memory of BMC."**

Case #1 – Malicious BIOS Loaded Hypervisor (5)

Where MBLH lives:



IPMI - Intelligent Platform Management Interface
BMC – Baseboard Management Controller

Case #1 – Malicious BIOS Loaded Hypervisor (6)

Notes about IPMI and BMC implementation:
- **IPMI is core platform for server local and remote management**,
- **BMC is core controller in IPMI implementation** and has its own BIOS
- As of the time of Case #1 research, BMC was integrated in so named South Bridge (I/O Controller Hub 631xESB/632xESB or ESB2)
- **There are three flash memory BIOS devices** – system, BMC and supporting I/O controllers
- **BMC BIOS software is encrypted and decrypted internally before execution**; that was found by the scientist in Intel documentation. Thus, it cannot be reverse engineered other that when it is decrypted in BMC RAM for execution.

-**BMC is connected to the system bus and thus having direct access to the system RAM**. **IPMI and BMC have complete control of the computer** including power cycling and rebooting.
- By Russian law, encryption of more than 40 bits (256 bit in BMC) is prohibited in imported equipment.

Case #1 – Malicious BIOS Loaded Hypervisor (7)

When MBLH got in BMC BIOS:

Quote: “ … in new series of server boards … **there are "add-on" programs in flash memory working with BMC and executed on the CPU, and these programs work with CPU hardware virtualization level.**

… **Images of flash memory software from Intel's website do not contain such software modules, thus software modules preventing my hyper-driver from working properly were illegally embedded in the motherboard flash memory during the production stage**.”

Comment: This is, in general, the author’s conclusion concerning technical side of the first phase of his research. His conclusion about **“ … were illegally embedded … during the production stage” is based on the believe in “Assembled in China” label.** ***However, such label does not mean that BIOS alteration could not be done AFTER such boards left China manufacturing facility, and outside of China.***

Case #1 – Malicious BIOS Loaded Hypervisor (8)

Here we are – the first discussion of the concern in Russia:

Quote: "... **I informed the management of Kraftway about the problem** with the BMC flash memory firmware and questionable, in legal terms, situation with the new Intel chipsets and **received rather expected response in the style of "no buzzing, that may hurt business.“**

Improvement of MBHL:

" ... however, continuing regularly run their system on new series of motherboard shipments from China, and new samples. **All samples continued to work steadily. When I switched to the Chinese boards, I saw more and more miracles. It seemed that colleagues from abroad are actively improving the operations of their root hypervisor. Last shipments of suspicious boards behaved almost normally.**"

Bingo:

"... it happened what I was waiting for a long time - **motherboards from new shipment did not hang my hyper-driver at all.** I was in doubt if I was paranoid in my suspicions, but new case has strengthened my opinion."

Case #1 – Malicious BIOS Loaded Hypervisor (9)

New virtualization feature – new outcome of MBHL case:

"… I started to work when it was revision 7 … in revision 11 conditions of entry in virtual host for hardware virtualization have been improved significantly … **I added new features of the hardware virtualization in my hyper-driver, and installed new chip set in standard boards from China, in which everything had worked without any problems ... New features of the equipment did not work**.

I, resuming the experiments, **moved new revision 11 chip set to Canadian sample motherboards. And new features started working!**

… there is only one conclusion possible that **colleagues from abroad did not know about the new revision of hardware virtualization**."

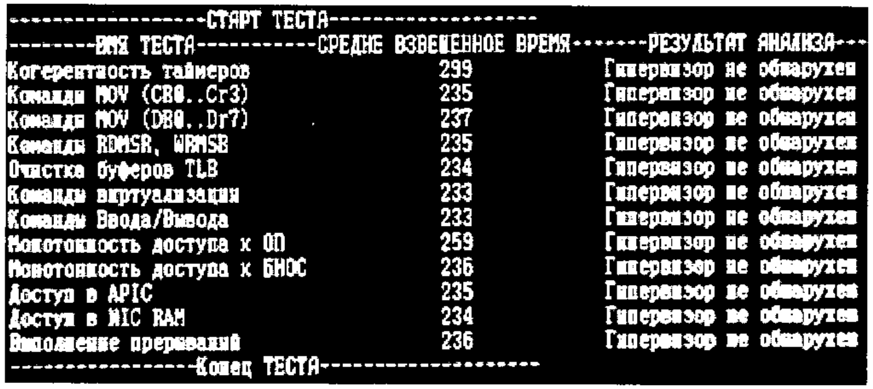
Final status:

"Already knowing how to deal with this problem, I downloaded in the standard motherboard firmware for the BMC from Intel site. I was sure that everything will work, but to my great surprise, it did not. Hanging was still there. This was something new."

Basically**, it was "freeze of the code" – the end of the development, ready for production!**

Case #1 – Malicious BIOS Loaded Hypervisor (10)

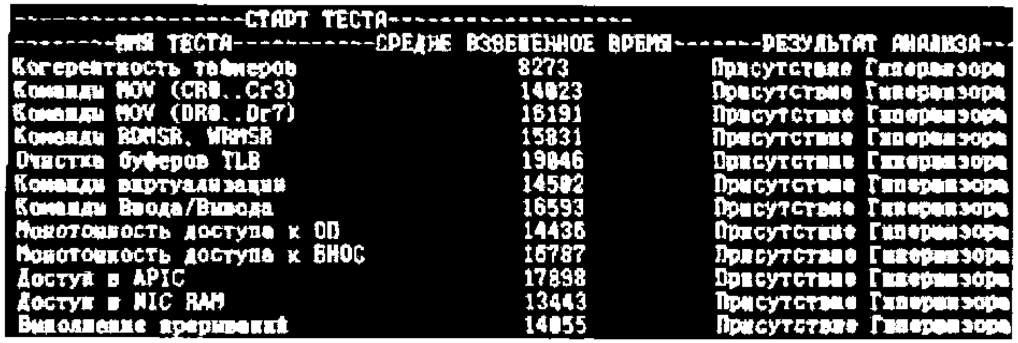
Possible MBLH developer?
All boards – both testing and production came from the same source – Kraftway company. **How boards “Assembled in China” came in Kfaftway is not known, and may be not from China at all. The company is, by the way, one of leading IT companies in Russia**, and thus is capable of high level software development as well.
New method of catching MBLH:
Fixing by downloading new BIOS revision did not work anymore, and the scientist **decided to measure execution time of system commands in boards from “China” and “Canada”**. Below is the result “**without MBLH**”:



```
-------------------СТАРТ ТЕСТА--------------------
---------ИМЯ ТЕСТА------------СРЕДНЕ ВЗВЕШЕННОЕ ВРЕМЯ-------РЕЗУЛЬТАТ АНАЛИЗА---
Когерентность таймеров                    299          Гипервизор не обнаружен
Команды MOV (CB0..Cr3)                    235          Гипервизор не обнаружен
Команды MOV (DB0..Dr7)                    237          Гипервизор не обнаружен
Команды RDMSR, WRMSB                      235          Гипервизор не обнаружен
Очистка буферов TLB                       234          Гипервизор не обнаружен
Команды виртуализации                     233          Гипервизор не обнаружен
Команды Ввода/Вывода                      233          Гипервизор не обнаружен
Монотонность доступа к ОП                 259          Гипервизор не обнаружен
Монотонность доступа к БИОС               236          Гипервизор не обнаружен
Доступ в APIC                             235          Гипервизор не обнаружен
Доступ в NIC RAM                          234          Гипервизор не обнаружен
Выполнение прерываний                     236          Гипервизор не обнаружен
------------------Конец ТЕСТА---------------------
```

Case #1 – Malicious BIOS Loaded Hypervisor (11)

The second part of the experiment – with MBLH:



```
-------------------СТАРТ ТЕСТА--------------------
---------ИМЯ ТЕСТА------------СРЕДНЕ ВЗВЕШЕННОЕ ВРЕМЯ-------РЕЗУЛЬТАТ АНАЛИЗА---
Когерентность таймеров                    8273            Присутствие Гипервизора
Команды MOV (CR0..Cr3)                    14023           Присутствие Гипервизора
Команды MOV (DR0..Dr7)                    16191           Присутствие Гипервизора
Команды RDMSR, WRMSR                      15831           Присутствие Гипервизора
Очистка буферов TLB                       19046           Присутствие Гипервизора
Команды виртуализации                     14502           Присутствие Гипервизора
Команды Ввода/Вывода                      16593           Присутствие Гипервизора
Монотонность доступа к ОП                 14436           Присутствие Гипервизора
Монотонность доступа к БИОС               16787           Присутствие Гипервизора
Доступ в APIC                             17898           Присутствие Гипервизора
Доступ в NIC RAM                          13443           Присутствие Гипервизора
Выполнение прерываний                     14055           Присутствие Гипервизора
```

The Result of Analysis column says “**Hypervisor is detected**”. **The execution time has indeed significantly increased, but it is approximately 60 times more for all tested commands!** It is a bit hard to believe that MBLH software utilizing hardware virtualization increased execution time of some commands 60 times!

Case #1 – Malicious BIOS Loaded Hypervisor (11)

Expressing concerns and presenting the case to various parties;

1. **Intel representative in Russia** – no response
2. **FSB (Federal Security Services**, former KGB) “Center for Information Protection” (CIP) mid-level executives and specialists – misunderstanding and almost no interest expressed.
3. **FSB CIP second meeting** – senior level executives and specialists - The meeting turned into a lecture. … and finally answered many questions. At the end of the meeting they thanked us, and said that the issue should be investigated in the framework of special research.” Finally – the case is dismissed.
4. **To prove the case** “…… I should write such a "add-on" myself. **I would not be able to put the "add-on" in the flash memory of the BMC, but can load the code in the main BIOS.** … My Last Judgment Day weapon **will utilize the "add-on" to kill the computer system by an external command** … actually perform single function task of wiping out BIOS flash chip when receiving a command to destroy the system.” **Finally he managed to show the demo to GasProm (state oil and gas enterprise0) security team**, but with the same success as to others. ***Dismissed.***

Case #1 – Malicious BIOS Loaded Hypervisor (12) Conclusion

1. **The post describes unique experience of identifying a malicious hypervisor, which is loaded before OS bootup** and right after system BIOS hardware initialization.
2. The post definitely raises some questions, because important technical information concerning experiments is missed. However, considering the post contents, **we give it 90% of chance being reality while leaving 10% to the benefit of doubt.**
3. The author identified, while missing details, **that the malicious hypervisor is embedded in BMC BIOS.** Last years' research on modern exploits of IPMI and its weaknesses completely correlates with that.
4. The author describes his experiments and the **development and improvement of MBLH up to the level when it cannot be identified** even by running yet another virtualization software on the same level.
5. **Experiments with commands' execution time to identify the existence of MBLH are important but lacking details**, which would lower our suspicion level.
6. The author's ordeal while trying to convince authorities and technical professionals is very typical and well known to security professionals.
7. While **the author believes that there is "Chinese Hand"** in altering BMC BIOS, **we have some doubts, because the label "Assembled in China" does not mean anything,** like the label "Assembled in Canada" where we did not find Intel facility related to manufacturing and assembly process. ***By details of this case, MBLH might be developed by Kraftway or FSB.***

Case #2 – the US – Is where the Idea of Malicious Hypervisor Came from?

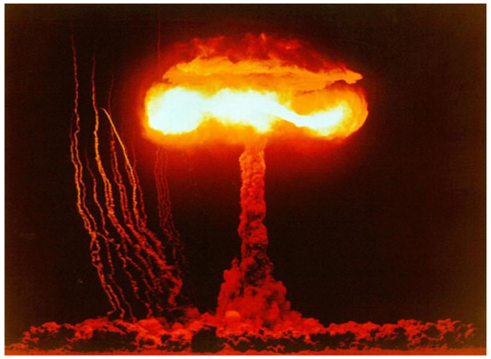
Not too much known research with prominent sponsors:
Joined team from University of Michigan and Microsoft Research did sponsored research “SubVirt: Implementing Malware with Virtual Machines” which (quote) “was supported in part by **National Science Foundation** grants CCR-0098229 and CCR-0219085, by **ARDA** grant NBCHC030104, by **Intel Corporation**, and by **Microsoft**”.
What is ARDA? Would then ARDA be now DARPA? Yes, many thanks to Wikipedia, **ARDA = DARPA (Defense Advanced Research Projects Agency)**.
Therefore, a while before Russian scientist started fighting with a ghost embedded in BMC BIOS, the foundation for malicious hypervisor has been researched, and two proof-of-concepts implemented sometime in 2005 – 2006 years.
This research is very important to understand various aspects of now existing problem of likely existing invisible malware and perspectives to mitigate that.

Case #2 - The Purpose of VMBR Research

Quote: **“We evaluate a new type of malicious software that gains qualitatively more control over a system. This new type of malware, which we call a virtual-machine based rootkit (VMBR), installs a virtual-machine monitor underneath an existing operating system and hoists the original operating system into a virtual machine.** Virtual-machine based rootkits are hard to detect and remove because their state cannot be accessed by software running in the target system.”
***VMBR is different from MBLH, because it is not embedded in BMC BIOS but loaded in RAM, and thus has different functionality.* Just the first step …**
Quote: “**Our project which we called SubVirt, shows how attackers can use virtual-machine technology to address the limitations of current malware and rootkits.**
…We show how attackers can install a virtual-machine monitor (VMM) (comment: hypervisor) underneath an existing operating system and use that VMM to host arbitrary malicious software. The resulting malware, which we call a **virtual machine based rootkit (VMBR), exercises qualitatively more control than current malware, supports general purpose functionality, yet can completely hide all its state and activity.**”

Case #2 - Details of Ultimate Weapon Research

The research goals:
"**We demonstrate that a VMBR can be implemented on commodity hardware and can be used to implement a wide range of malicious services**."
"We show that, once installed, a VMBR is difficult to detect or remove."
"**We implement proof-of- concept VMBRs on two platforms** (Linux/VMware and Windows/VirtualPC) **and write malicious services** such as a keystroke sniffer, a phishing web server, a tool that searches a user's file system for sensitive data, and a detection countermeasure which defeats a common VMM detection technique."
"**Finally, we discuss how to detect and defend against the threat posed by VMBRs** and ***we implement a defense strategy suitable for protecting systems against this threat***."



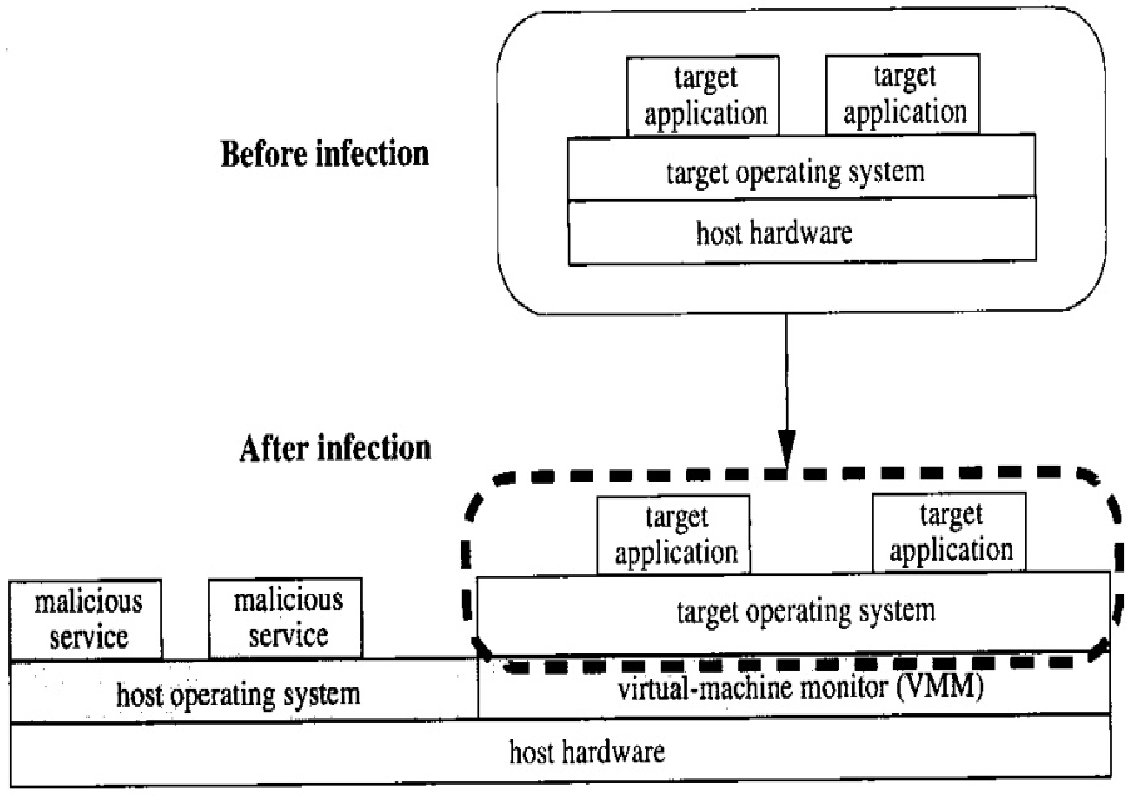
Case #2 – How to Control Guest OS

In "normal" virtual machine implementation there is no need for host OS to completely control guest OS. However, VMBR requires that:
" … **One problem faced by VM services is the difficulty in understanding the states and events inside the guest they are serving**; **VM services operate at a different level of abstraction from guest software.** Software running outside of a virtual machine views low level virtual-machine state such as disk blocks, network packets, and memory. Software inside the virtual machine interprets this state as high-level abstractions such as files, TCP connections, and variables. **This gap between the VMM's view of data/events and guest software's view of data/events is called the semantic gap**."
"Virtual-machine introspection (VMI) describes a family of techniques that enables a VM service to understand and modify states and events within the guest. VMI translates variables and guest memory addresses by reading the guest OS and applications' symbol tables and page tables. VMI uses hardware or software breakpoints to enable a VM service to gain control at specific instruction addresses. "

Case #2 – Design and Implementation Notes (1)

VMBR installation process (quote):
“In the overall structure of a VMBR, a VMBR runs beneath the existing (target) operating system and its applications. To accomplish this, a **VMBR must insert itself beneath the target operating system and run the target OS as a guest.** To insert itself beneath an existing system, a **VMBR must manipulate the system boot sequence to ensure that the VMBR loads before the target operating system and applications**. After the VMBR loads, it boots the target OS using the VMM. As a result, the target OS runs normally, but the VMBR sits silently beneath it.” ***This process is different from MBLH where the hypervisor is loaded from BMC and then boots the OS***.
Malicious services (quote):
“**Using our proof-of-concept VMBRs, we developed four malicious services that represent a range of services** a writer of malicious software may want to deploy. We implemented a phishing web server, a keystroke logger, a service that scans the target file system looking for sensitive files, and a defense countermeasure that defeats a current virtual-machine detector. To develop these services, we use the host OS as our attack OS (below):

Case #2 – Design and Implementation Notes (2)



Case #2 – Design and Implementation Notes (3)

Malicious services:
The authors of this research definitely went much further than we originally discussed as malicious hypervisor and MBLH. They also introduced exploits on the top of malicious hypervisor. That is actually going to the very end of a system compromise.
Maintaining control – defending VMBR:
Quote: " …**The only time the VMBR loses control of the system is in the period of time after the system powers up until the VMBR starts**. if the BIOS boots a program on an alternative medium, that program can access the VMBR's state."
There are two measures to regain the control:
- **Handling reboots by restarting the virtual hardware rather than resetting the underlying physical hardware** thus creating an illusion of reboot.
- **Emulation of system shutdown by using ACPI (Advanced Configuration and Power Interface) sleep states to switch hardware in lower power mode**, **which look s like a shutdown while memory state is intact**. The most targeted systems like ***servers are really rear shutdown***.

Case #2 – Design and Implementation Notes (4)

Evaluation of the implementation:
Quote: “This section evaluates the impact of a VMBR on a system. We valuate the disk space used by a VMBR, the time to install a VMBR, the effect of a VMBR on the time to boot the target OS, the impact of a VMBR as viewed by a user, and the effect of the memory space used by a VMBR.”
Comment:
The section contains interesting technical material, and could be helpful in further research of MBLH case. **However, the VMBR research has been done on systems not yet supporting hardware virtualization, so dependences and numbers will be completely different for modern systems starting from 2007 year like in MBLH case.**

Case #2 - Defending against VMBR (1)

This is possibly the most interesting part of this research – **how the creators of ultimate weapon suggest protecting against it.** Quote: “In this section, we explore techniques that can be used to detect the presence of a VMBR. VMBRs are fundamentally more difficult to detect than traditional malware because they virtualize the state seen by the target system and because an ideal VMBR modifies no state inside the target system. “

There are two methods:

Security software *below* the VMBR:

1. **To run a “detector” software below VMBR** – **by the VMBR concept, only system BIOS is below, so this recommendation is not concrete.**
2. **Secure boot prevents loading drivers and other software modules not having a “signature”, which compared with “platform key” in a firmware**. However, that is the function of boot loader to compare the key and signatures, but **one of the main functions of VMBR is the loader alteration**. **Secure boot is actually above VMBR, not below**.
3. **Other methods are secure hardware, the boot from save media and secure virtual machine monitor.** All of them will try to boot by other than affected hard drive. However, **we’ve seen a malware which blocks any boot but the hard drive** and does not matter what is configured in BIOS.

Case #2 - Defending against VMBR (2)

Secure Boot – continue:
“Using a secure VMM, we implemented an enhanced version of secure boot which can prevent VMBR installations. The goal of our secure boot system is to provide attestation for existing boot components, such as the disk’s master boot record, the file system’s boot sector, and the OS’s boot loader and also to allow legitimate updates of these components. All attempted updates of these components are verified (by checking the cryptographic signature) before they are allowed to complete.”
Our opinion:
We see some inconsistency in this concept. When such “secure VMM is installed? If before VMBR, that each computer system requires such component. If after VMBR is installed first,it controls any other installations, thus can prevent creating VMM on its own level. **We, in general, disagree with that anything could be installed below VMBR, because it is installed right after BIOS hardware initialization and initial tests, and thus can control any other installation.**

Case #2 - Defending against VMBR (3)

Software above VMBR - many obstacles:
1. **General obstacle to run security software above VMBR is, as discussed, its position above** and thus it is completely controlled by VMBR. **However, if VMBR utilizes all computer hardware resources, then (quote) "… timing differences can be noticed by software running in the virtual machine by comparing the running time of benchmarks against wall-clock time**".
The overhead in CPU, memory, hard drive could be measured, ***and then may be altered by VMBR as well. However, external clock will deliver unaltered time.***
We would like to note here that ***such measurements require two computer states – before and after VMBR installation*, *or two identical systems to compare*, *or somehow disabling VMBR***. We will discuss our experiments and conditions of "overhead" testing below.
2. Another option is utilization of HIDS system which would test installed software , for instance, I/O drivers for changes which required for virtualization.

Case #2 - Conclusion

Case #2 goals:

- The design and implementation of Virtual Machine Based Rootkit, Implementation of malware utilizing VMBR features, Evaluation of VMBR, Implementation on two platforms, Detection and defense against VMBR.

The research and the paper representing it are perfectly done, and convincing that all practical implementations worked according to the research theory and goals.

Our opinion:

Authors finally tried to downgrade potential threat of VMBR-like systems and leverage a possibility to overcome it.

We do not share authors' pessimism over VMBR deadly capabilities and optimism over detection and defeat. "Russian Ghost" Case #1 perfectly correlates with VMBR concept, and goes further by exploiting BMC capability to run malicious hypervisor on the lowest level possible. We believe that such hypervisor could even run before system BIOS thus virtualizing it and its functions.

By our opinion, there are two main results – the proof-of-concept and the code of VBMR. ***The research has been done before IEEE symposium in May, 2006. Considering that the research described in "Russian Ghost"" Case #1 began in 2007 or later, there is complete time correlation. The concept and the code could be used to move VMBR to the next level – BMC BIOS based MBLH. The threat level thus has significantly increased.***

Our Case – Global or Local Threat and Possible Implementation

The research in the Case #2 lowered our doubts that the Case #1 is a myth. *We see definite correlation and the development of initial idea of VMBR. However, even if Case #1 were complete myth, sooner or later the idea of VMBR would find its supporters and implementers in the form of MBLH.*

In the following paragraph we would like to identify the place and the vector of VMBR/MBLH threat. We will do our analysis bases on two already discussed cases first, and then will review our conclusion considering the Case #3, which follows.

Global or local threat? That is may be the most important question. It depends on how easy is to deploy and to distribute the threat.

Our case – VMBR and MBLH Deployment

Case #2 (VMBR) does not consider any particular deployment rather than RAM, while Case #1 (MBLH) pinpoints BMC BIOS. BMC BIOS has its own firmware file, which is to be deployed together with others.
The entire BOIS file is signed by Intel QA, and downloaded from authorized resource. Thus, **it is very unlikely that official Intel BIOS file contains an alteration, and more likely it happened after motherboard manufacturing, or server assembly process**.
We think that alteration process with associated logistics is very manual and is done for particular targets. We do not see a possibility of mass production of altered board to deploy around the globe.
So, it is very unlikely that a maniac hijacked Intel facility and spreads altered software with the purpose of pressing a button "Die All". However, **the access to data sources is very valuable in some cases.**

Our deployment conclusion: it is more feasible to alter some motherboards or servers targeting highly valuable data resources. Thus we dismiss the global option and incline to consider local MBLH deployments

Our case – VMBR and MBLH Distribution

Distribution:
Computing resources with MBLH will be physically shipped to targets as the first part of infiltration process. Then, utilizing MBLH resources and local network vulnerabilities other servers could be hijacked and converted to silent and invisible bots. Such process of searching and converting local targets does not require external resources excepting sites for downloading altered BIOS versions.
Our Case conclusion:
1. So far, based on what we know by two cases, **we do not expect worldwide distribution of MBLH infected computers because of high complexity of BMC BIOS alteration process and deployment**. Targeted "customers" may receive a shipment with altered BMC BIOS, and thus entire local network could be eventually hijacked.
2. **Concerning uploading VMBR in a computer system RAM, Case #2 research did not really consider options**. That could be local malicious activity of IT personnel uploading VMBR from a CD-ROM, etc. **In any case, local uploading is really limited as means of VMBR distribution.**

Case #3 – Widespread Distribution of Malicious Hypervisor via IPMI vulnerability

Now is the Case #3 - we return to Case #1 but with different perspective. Michigan University research **“Illuminating the Security Issues Surrounding Lights-Out Server Management”** by Anthony J. Bonkoski, Russ Bielawski and J. Alex Halderman. Their research gave us new information applicable to already considered cases.
Quote: “**This paper examines the security implications of the Intelligent Platform Management Interface (IPMI), which is implemented on server motherboards using an embedded Baseboard Management Controller (BMC).** We consider the threats posed by an incorrectly implemented IPMI and present evidence that IPMI vulnerabilities may be widespread. “
“**We analyze a major OEM’s IPMI implementation and discover that it is riddled with textbook vulnerabilities, some of which would allow a remote attacker to gain root access to the BMC** and potentially take control of the host system.
Using data from Internet-wide scans, we find that there are at least 100,000 IPMI-enabled servers (across three large vendors) running on publicly accessible IP addresses, contrary to recommended best practice”.

Case #3 – Introduction and Related Topics

IPMI: Intelligent Platform Management Interface has been known since 1998, is independent from OS and functions even when computer system is down. **For yet unknown reason, up until 2012 – 2013 IPMI insecurity was not widely discussed, with the exception of our Case #1**, **when Russian scientist pointed out to BMC embedded malicious hypervisor.**
The research lists various known exploits, which could be installed in BMC.
Quote: **"Malware residing on the BMC could be extremely difficult to detect, since it sits at an even lower architectural layer than a BIOS or VM-based rootkit (reference to VMBR)".**
It is considered as common sense security practice not to connect IPMI devices (i.e. its network controller) to public network. However (quote)"… we use data from Internet-wide surveys **to reveal public IP addresses of over 100,000 IPMI devices, including more than 40,000 systems that our results suggest are remotely exploitable."**
Quote:"…Instead of securing IPMI vendors do rather opposite: "**The vulnerabilities we find, along with others previously found …, suggest that some IPMI manufacturers are systematically failing to properly secure these devices."**

Case #3 – IPMI Security Risks

All known BMC OS use Linux OS which as any requires security updates. "IPMI malware carries similar threats (like BIOS) and is likely easier to develop, since many BMCs run a standard operating system. BMC malware would also likely be easier to install remotely, due to IPMI's substantial network-facing attack surface."

Attack surface: IPMI (Supermicro Inc. implementation) has six TCP and UDP ports for communication and management. **Even if recommended separate management network is used, there still is an opportunity to connect to BMC exploiting various issues**.

Authentication risks: Administrators tend to using default or the same password on multiple servers. And IPMI device may use insecure password storage. Thus, after compromising one system it is possible to get access to multiple.

Case #3 – Attack Scenarios

1. Subverting the host system- by exploiting remote management capabilities and thus accessing the most of server configuration
2. BMC spyware against host's OS– Quote: "If the attacker can install malware on the BMC, it would have a powerful vantage point for spying on the system and its administrator."
3. Persistent attack from BMC – Quote: "As the BMC operates independently from the host's operating system and CPU, it provides an ideal **hiding place for a stealthy, highly persistent rootkit….A BMC rootkit would survive reinstallation of the host's OS, or even complete replacement of the host's storage devices. … could even be designed to survive BMC firmware updates by dynamically patching the new firmware**."
4. Attacking BMC from the host system – an attacker could re-flash the BMC's firmware from compromised host's OS.
5. IPMI botnets – Quote: "If widely used IPMI devices can be compromised remotely, they can be leveraged to create large networks of bots. … **the system operator is unable to run normal malware detection and removal tools within the BMC.**"
We see that 4 out of 5 scenarios tightly correlate with the Case #1 claim that malicious hypervisor had been embedded in BMC firmware.

Case #3 – Analysis, Attacks and Network measurements

Quote: “To explore the potential for BMC compromise, **we analyzed an IPMI implementation shipped by one large server manufacturer, Supermicro**. … **We ultimately discovered a range of vulnerabilities, and we developed two proof-of-concept exploits to demonstrate some of the most critical problems**.”

Vulnerabilities and associated attacks:

Insecure input validation, Shell injection vulnerabilities, Buffer overflow vulnerabilities and Buffer overflow exploit.

(Quote) **“ … These vulnerable firmware images apply to 135 Supermicro product models. The problems may also affect IPMI devices from other manufacturers that are based on similar ATEN firmware.”**

Network measurements:

Quote: “… we used data from an Internet-wide network survey conducted in May 2013”. **While the search was limited to certificates’ analysis, the authors were able to identify over 105,000 servers of major manufacturers (Dell, Supermicro and HP) connected to Internet, and likely having vulnerabilities**.

Case #3 - The research conclusion

Quotes:
“… Since BMCs operate independently of the host system and CPU, cleverly written malware running there *could potentially reside undetected indefinitely*. Unfortunately, due to the closed nature of BMC firmware, server operators have few avenues to defend themselves without vendor assistance.”
” … We uncovered a wide range of vulnerabilities and demonstrated two working attacks that allowed us to gain root shell access. These problems pose an immediate threat to many systems in the field; we found over 40,000 devices similar to the one we analyzed visible on public IP addresses.”
“…In the long run, securing remote management systems calls for a defense-in depth approach. Vendors need to apply careful security engineering practices, minimize attack surfaces, and help users ensure that their systems are appropriately locked down and isolated from public networks.”

Our Case #3 conclusion

1. The research radically changes our preliminary conclusion given in p.4.2. Now, combining all three cases, **we see global threat by VMBR/MBLH distribution via IPMI vulnerability**, which is the combination of very low security of IPMI/BMC implementations and human factor – simply security ignorance of IT personnel. Numbers shown in the Case #3 research are not final, and may turn in to millions of ready for exploits servers.
2. **We do not think that the situation will improve in near future, because of costs of IPMI/BMC securing and required time.** Neither we are optimistic over the “human factor” and IT attitude.
3. **We see that modern trend toward “cloud services” may affect overall information security.** Users of in most cases do not understand that “clouds” are simply monstrous datacenters with thousands of computers, and all of them serve users over Internet. **Having thousands of opened Internet connections, even with all security measures, represent high risk of one exploited and then all infected over shared vulnerable management network**. Needless to say that such centers are very likely to have management systems Internet connected as well.

Our Ten Cents to How to Identify MBLH Presence

We share the Case #3 opinion that identification from “below” simply does not exist as VMBR/MBLH is always on the lowest level, for instance embedded in BMC/IPMI.

Catching by 100% utilization and in “statistical net”:

1. **Instead of measuring commands’ execution time (Case #1), use 100% utilization** (Case #2) and watch on software execution time.
2. **We need to get two testing options – with or without MBLH. It is possible only if computer system BIOS supports turning hardware virtualization on or off.** Not all chip and server vendors support that. ***Our Dell T105 AMD Opteron based servers have virtualization always up – no BIOS settings for***.
3. **However, hardware virtualization will work if MBLH is not designed to reside below system BIOS**. In this case it will be able to alter BIOS function of turning virtualization off.
4. **We used supporting virtualization and BIOS settings Lenovo notebook** with Linux CentOS 6.6 OS installed on.
5. **This computer system BIOS has two virtualization related options** - Intel Virtualization Technology (Disable/Enable) and Intel VT-d Feature (virtualization technology support for direct I/O, Disable/Enable).

Our Ten Cents – Measurements and Statistical Net

No Virt.				**Virt.**			
Test #	Time	Dev	(Dev)2	Test #	Time	Dev	(Dev)2
31	19:54	1	1	41	19:57	5	25
32	19:49	-4	16	42	19:45	-7	49
33	19:55	2	4	43	19:50	-2	4
34	19:59	6	36	44	19:48	-4	16
35	19:46	-7	49	45	19:54	2	4
36	19:43	-10	100	46	19:40	-12	144
37	19:58	5	25	47	20:03	11	121
38	19:55	2	4	48	19:57	5	25
39	19:50	-3	9	49	19:59	7	49
40	20:00	7	49	50	19:49	-3	9
Mean:	19:53	Sum:	293	Mean:	19:52	Sum:	446
SD			5.4			SD	6.7

Our Ten Cents – Measurements and Statistical Net (2)

Our testing software "dowork" (it is available for public) does **floating point calculation driving CPU to 100% utilization**. Each calculation cycle takes a few seconds. **We used 400 cycles to get statistically more stable results**. However, various OS processes created certain deviation, thus **we used 10 tests to improve statistics in both cases when two virtualization options are disabled and when enabled**.
Our results presented in the following table where first four columns are for the test with disabled virtualization and next four – with virtualization; Dev is deviation, SD – standard deviation, time is "minutes:seconds" format.
We see that for both tests mean values are very close – 19:53 and 19:52. Standard deviation in the case of enabled virtualization is higher, but **statistically both results are very close – SD is 5.4 and 6.7.**

We did not find - statistically – the presence of a malicious hypervisor.

We think that such statistical testing approach on long running programs is the only one way to overcome live OS variations in execution time and is likely to identify the presence of a malicious hypervisor.

Our Ten cents – External MBLH Catching - SIEM

There is one more potential opportunity to catch VMBR/MBLH. Modern Security Information and Event Management System (a SIEM is not a luxury anymore, but security necessity) may help to identify short and infrequent connections of a malicious hypervisor via management or system network interface to internal or external hosts. Basically, we are facing to find a needle in a hay stock, but this a SIEM is for – logging random and seldom events across entire network and systems and finding suspicious connections. We did not implement such hunt simply because we do not have servers with hardware virtualization support disabling functionality in system BIOS, but we encourage the audience not to ignore such opportunity if you have a few servers to experiment with and a SIEM in place.

Our research Conclusion

1. Case #1 – Russian mythical post. It is the only one but yet circumstantial evidence of successful attempt to embed a malicious hypervisor in BMC BIOS software. We consider it as “reality”, but give 10% to the benefit of doubt. **Its reference to BMC as the platform to run a malicious hypervisor completely correlates with two other cases.** The post author observed for long time continuing improvement of MBLH software until it became completely invisible to the second hypervisor. We used proposed test of execution in our attempt to find MBLH in our Lenovo notebook. **There is neither real evidence nor logic to suspect that MBLH has been embedded during manufacturing process in China.** May be later … at any place in the world.

2. Case #2 – US – Michigan University research ended approximately in 2006. Virtual Machine Based Rootkit (VMBR) is the fundamental research proving that malicious hypervisor can be developed and work on lower that OS level, and can move successfully OS one level up and replace it by itself occupying the lowest system level. **We think that this concept and may be the code itself have been used in the Case #1. We believe that the code has been almost publicly available and finally has been developed in Malicious BIOS Loaded Hypervisor (MBLH).**

The research authors were trying to downplay the danger of the solution and proposed two possible ways to identify and protect against it. **However, the method “from below” simply does not exist, and as case #1 showed, the hiding capability of VMBR is extraordinary.**

Our research Conclusion (2)

3. Based on Case #1 and Case #2 we do not expect worldwide distribution of MBLH infected computing resources because of high complexity of BMC BIOS alteration process. Targeted "customers" may receive a shipment with altered BMC BIOS, and thus entire local network could be eventually hijacked utilizing 0-day attacks and updating original BIOS to altered version.
4. Case #3 - **The research on IPMI vulnerability**, which has been done at Michigan University and published in 2013, **has changed the VMBR/MBLH threat landscape significantly. The research showed that production servers have numerous vulnerabilities, which make BMC and then entire system easy to compromise target. Negligence and ignorance of IT personnel managing server platform created situation when hundreds of thousands servers around the globe are Internet connected and thus can be easy exploited**. We now see that the most likely scenario for VMBR/MBLH distribution is over vulnerable BMC implementations either Internet or LAN connected.
5. Cloud - The most dangerous situation, as we see that, is at "cloud" services datacenters, which have hundreds of servers working with users over Internet; it is possible that these servers' management system also connected to public network.
6. Even without Case #1, **it would be enough two MU research projects to get us to the same conclusion about worldwide threat of VMBR embedded in BMC and exploiting IPMI.**

Pure Summary

- Russian Ghost/VMBR/MBLH is dangerous as can infiltrate in millions of servers worldwide
- The hunting season is opened but be patient – we have only two perspective tools so far
- In theory, we cannot identify it, but we still have a chance …
- There is no protection against it – put your server is a dumpster – special thanks to IPMI
- ***NO security standard calls for server management (IPMI) protection***

References

1. Chinese Add-ons: True Stories of virtualization, information security and computer spying; post on http://xakep.ru/articles/58104/ 12/26/2011. Translated from Russian, Copyright © DeepSec, GmbH and Rubos, Inc., 2014.
2. Intelligent Platform Management Interface. Wikipedia, see http://en.wikipedia.org/wiki/Intelligent_Platform_Management_Interface
3. SubVirt: Implementing malware with virtual machines. Samuel T. King, Peter M. Chen (University of Michigan); Yi-Min Wang, Chad Verbowski, Helen J. Wang, Jacob D. Lorch (Microsoft Research); IEEE Symposium on Security and Privacy, Berkley/Oakland, CA, USA, 21-24 May, 2006.
4. Illuminating the Security Issues Surrounding Lights-Out Server Management by Anthony J. Bonkoski, Russ Bielawski, J. Alex Halderman; Michigan University. 7Th USENIX Workshop on Offensive Technologies, August 13, 2013, Washington, DC. https://www.usenix.org/conference/woot13/workshop-program/presentation/bonkoski
5. Unified Extensible Firmware Interface, Wikipedia; http://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface

Thank you!

All questions will be answered:

- mikhailutin@hotmail.com

or

- mutin@rubos.com

- This presentation will be available on DeepSec site

A Myth or Reality – BIOS-based Hypervisor Threat

Special research prepared by Rubos, Inc. team

(We do independent research on security matters in various domains)

Prepared for DeepSec 2014

Speaker: Mikhail Utin, PhD, CISSP

mikhailutin@hotmail.com

(Questions will be answered after the presentation. Please, submit them to the speaker in writing)

Myths and Reality

Myths and reality intersect and interchange.

Greek myth of Achilles – in modern terms is a “search of an ultimate protection and creation of single point of failure”

Fig.1. *Thetis Dipping*

Modern Myths

- Ghosts, witches, witchcraft; Ghost hunting have been entertaining but unsuccessful – none apprehended and caged for public exposition ; for believers- reality, for others – a myth
- UFO and UFOlogy – a sort of ghost hunting; believers have a fun and consider a reality, but yet a myth – no alien is available for public demo
- IT Management – some see as reality, but in cases like Case #3 below – just a myth
- Information Security – do we have an official myth or a ghost?
- **A myth about Malicious Hypervisor (Russian Ghost) appeared on Russian hackers' site at the end of 2011. It has all myth's attributes. There were rumors about the post, and the storyteller described it as reality. However, neither he nor somebody else got and caged the ghost for public exhibition**

Russian Ghost Myth – Case #1 and Following Research

We believe that it was real or may still exist, **and we possibly know where it was born and eventually escaped from.**

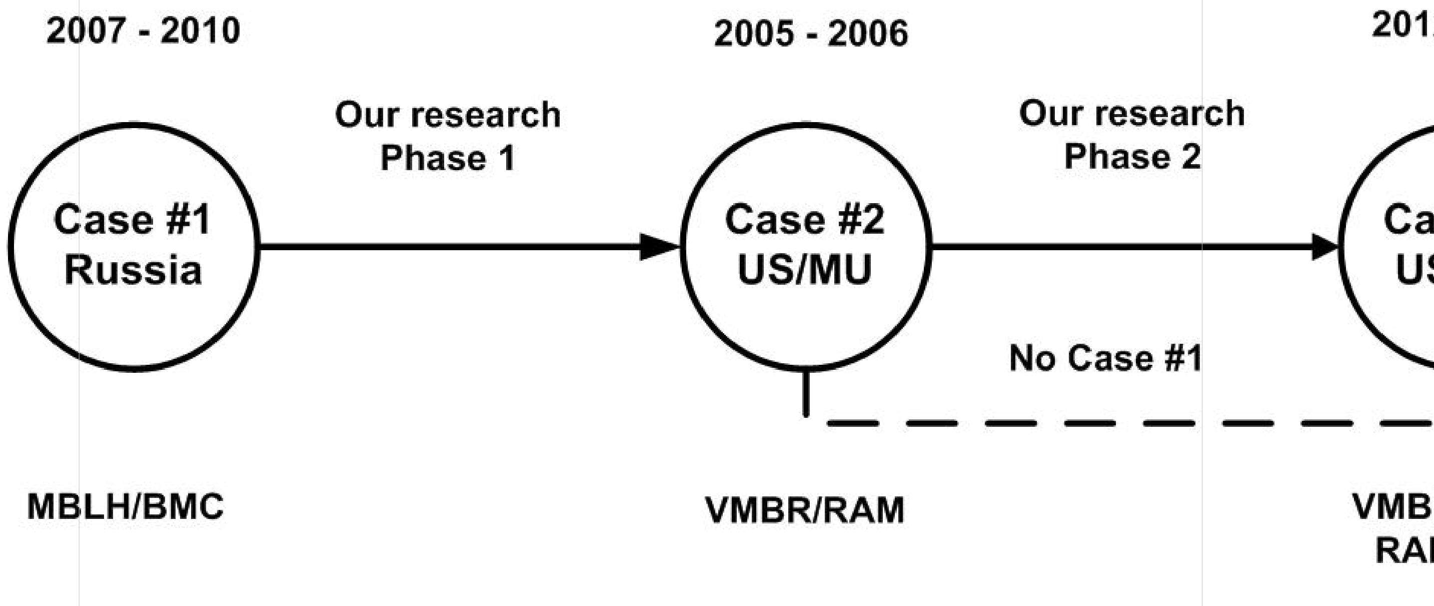
Our research is about three cases. We are interested to identify not only the "percent of reality", but also how dangerous it is, how to hunt on, and how to protect from:

Case #1 - ***Russian research on Malicious BIOS Loaded Hypervisor*** (approximately 2007 – 2010 years) posted at the end of 2011

Case #2 - ***US Michigan University Virtual-Machine Based Rootkit*** research which was done approximately in 2005 – 2006 years (published 2006)

Case #3 - ***US Michigan University IPMI/BMC (Intelligent Platform Management Interface/Baseboard Management Controller) vulnerability*** research of approximately 2012 – 2013 (published 2013) .

Research Direction



Case #1 - Russian Ghost – a Real Post and a Myth about Malicious BIOS Loaded Hypervisor (MBLH) (1)

- Was accidently found in the middle of 2012
- **Contains various topics (not technical only) and details**
- Has been published on 12/26/2011
- **Written in Russian and published on Russian hackers' site**
  If it were a scam with the intention of misleading, it would be published in English. **The fact that he post is almost unknown relates to its language**.

We translated the text and its summary is published on DeepSec site. Please, see it. Complete text will be available as well.

Reference: Chinese Add-ons: True Stories of virtualization, information security and computer spying; post on http://xakep.ru/articles/58104/ ; 12/26/2011

Translated from Russian, Copyright © DeepSec, GmbH and Rubos, Inc., 2014.

Case #1 – Malicious BIOS Loaded Hypervisor (2)

The project:
Typical Russian computer science project – to develop high performance computer system; no association with information security. **The customer – Kraftway, one of biggest Russian IT companies working closely with Russian government. It supplied Intel motherboards for the project.**
Quote: “I decided to use new Intel processors with the support of hardware virtualization even before the official announcement (in early 2007), and to create a unified computing system … **It required writing compact hypervisor with non-standard functionality. …It is essential to boot the OS on already virtualized platform, so even the first commands of the OS boot loader would be executed in a virtual environment**. Then the hypervisor has to virtualize the CPU real mode and all other operating modes. … it was overstatement to call that as "hypervisor", **I began to use the term "hyper-driver**”.
“ …**thanks to the cooperation with the company Kraftway I had access to pre-production models of processors and motherboards**…”
Comment: pre-production boards were from Kraftway company and labeled “Assembled in Canada”.. **They have been used for the design phase.**

Case #1 – Malicious BIOS Loaded Hypervisor (3)

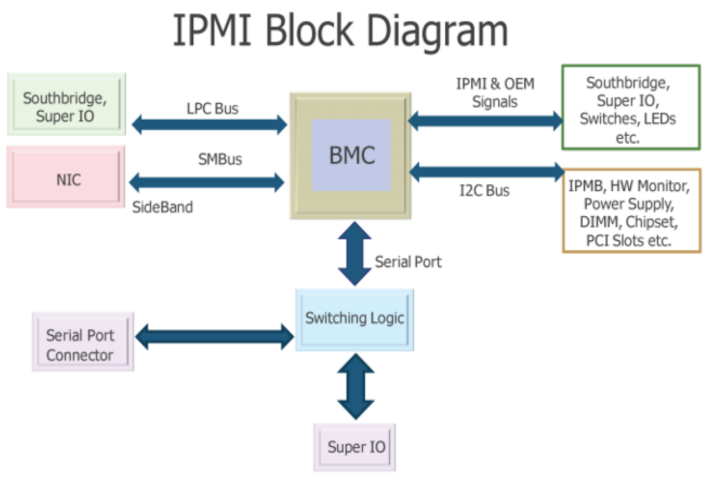
The problem:
"**I had to assemble a system based on first production motherboards**, which had just appeared. **However, the system did not work.** … I began to investigate and finally realized that my system hangs while executing hardware virtualization commands … **and finally get the BIOS code from the official Intel website and reloaded it in the motherboard, and then system started working.**"
The reason:
"**China-made motherboard started to work only when I uploaded the BIOS taken from Intel site.** … **It became clear that the boards from China contain additional software modules, embedded in the BIOS, and the standard analysis software does not see them**. Apparently, they also worked with hardware virtualization and thus were able to hide the true contents of the BIOS. … **Chinese boards had two software systems working simultaneously on the same hardware virtualization level, which does not allow sharing of resources**." By "two software" he means his "hyper-driver" and embedded malicious hypervisor.

Case #1 – Malicious BIOS Loaded Hypervisor (4)

Comment:
Here is very significant finding **– there is malicious hypervisor embedded in BIOS, and which utilizes hardware virtualization Intel CPU capability. It was possible to identify that because there was a conflict in sharing resources**. The developers of Malicious BIOS-Level Hypervisor (MBLH) did not expect that the system will run the second virtualization software on the same level. **We found official Intel documents that the company indeed has assembly and testing facility (CD1) in Chengdu, China opened in 2005, and another factory opened in Chengdu in 2007 (CD6). However, in Canada we found only Flash Memory group in Vancouver/BC, which is unlikely being involved in assembly process. Thus, there are some doubts concerning the label on the sample boards "Assembled in Canada".**
The home of the MBLH:
"… **it was necessary to begin to figure out in which of the flash chips was a software module that works with hardware virtualization**. … I was not even surprised when found out that **the "add-on" hypervisor starts working just after reloading software in flash memory of BMC."**

Case #1 – Malicious BIOS Loaded Hypervisor (5)

Where MBLH lives:



IPMI - Intelligent Platform Management Interface
BMC – Baseboard Management Controller

Case #1 – Malicious BIOS Loaded Hypervisor (6)

Notes about IPMI and BMC implementation:
- **IPMI is core platform for server local and remote management**,
- **BMC is core controller in IPMI implementation** and has its own BIOS
- As of the time of Case #1 research, BMC was integrated in so named South Bridge (I/O Controller Hub 631xESB/632xESB or ESB2)
- **There are three flash memory BIOS devices** – system, BMC and supporting I/O controllers
- **BMC BIOS software is encrypted and decrypted internally before execution**; that was found by the scientist in Intel documentation. Thus, it cannot be reverse engineered other that when it is decrypted in BMC RAM for execution.

-**BMC is connected to the system bus and thus having direct access to the system RAM**. **IPMI and BMC have complete control of the computer** including power cycling and rebooting.
- By Russian law, encryption of more than 40 bits (256 bit in BMC) is prohibited in imported equipment.

Case #1 – Malicious BIOS Loaded Hypervisor (7)

When MBLH got in BMC BIOS:

Quote: " … in new series of server boards … **there are "add-on" programs in flash memory working with BMC and executed on the CPU, and these programs work with CPU hardware virtualization level.**

… **Images of flash memory software from Intel's website do not contain such software modules, thus software modules preventing my hyper-driver from working properly were illegally embedded in the motherboard flash memory during the production stage**."

Comment: This is, in general, the author's conclusion concerning technical side of the first phase of his research. His conclusion about **" … were illegally embedded … during the production stage" is based on the believe in "Assembled in China" label.** ***However, such label does not mean that BIOS alteration could not be done AFTER such boards left China manufacturing facility, and outside of China*.**

Case #1 – Malicious BIOS Loaded Hypervisor (8)

Here we are – the first discussion of the concern in Russia:

Quote: "… **I informed the management of Kraftway about the problem** with the BMC flash memory firmware and questionable, in legal terms, situation with the new Intel chipsets and **received rather expected response in the style of "no buzzing, that may hurt business."**

Improvement of MBHL:

" … however, continuing regularly run their system on new series of motherboard shipments from China, and new samples. **All samples continued to work steadily. When I switched to the Chinese boards, I saw more and more miracles. It seemed that colleagues from abroad are actively improving the operations of their root hypervisor. Last shipments of suspicious boards behaved almost normally.**"

Bingo:

"… it happened what I was waiting for a long time - **motherboards from new shipment did not hang my hyper-driver at all.** I was in doubt if I was paranoid in my suspicions, but new case has strengthened my opinion."

Case #1 – Malicious BIOS Loaded Hypervisor (9)

New virtualization feature – new outcome of MBHL case:

"… I started to work when it was revision 7 … in revision 11 conditions of entry in virtual host for hardware virtualization have been improved significantly … **I added new features of the hardware virtualization in my hyper-driver, and installed new chip set in standard boards from China, in which everything had worked without any problems … New features of the equipment did not work**.

I, resuming the experiments, **moved new revision 11 chip set to Canadian sample motherboards. And new features started working!**

… there is only one conclusion possible that **colleagues from abroad did not know about the new revision of hardware virtualization**."

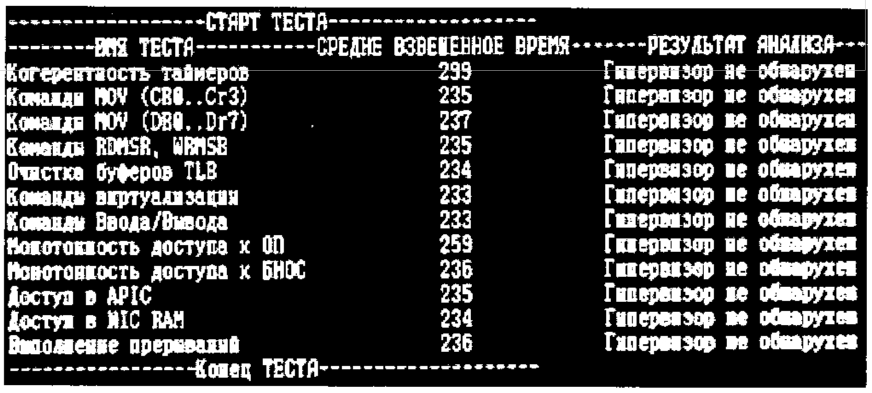
Final status:

"Already knowing how to deal with this problem, I downloaded in the standard motherboard firmware for the BMC from Intel site. I was sure that everything will work, but to my great surprise, it did not. Hanging was still there. This was something new."

Basically**, it was "freeze of the code" – the end of the development, ready for production!**

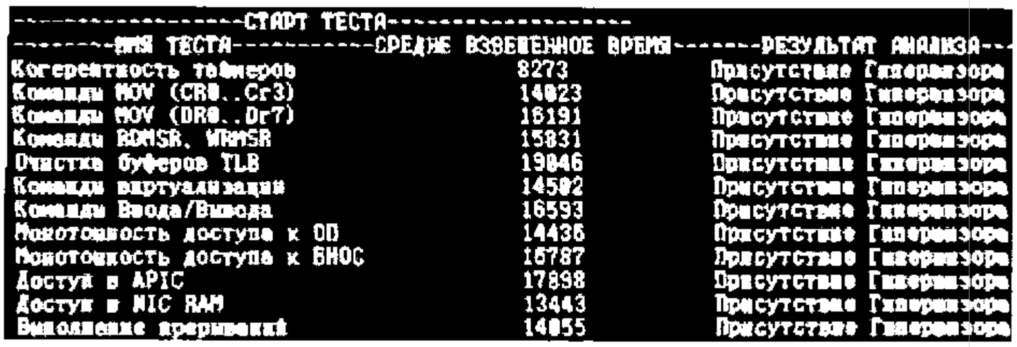
Case #1 – Malicious BIOS Loaded Hypervisor (10)

Possible MBLH developer?
All boards – both testing and production came from the same source – Kraftway company. **How boards “Assembled in China” came in Kfaftway is not known, and may be not from China at all. The company is, by the way, one of leading IT companies in Russia**, and thus is capable of high level software development as well.
New method of catching MBLH:
Fixing by downloading new BIOS revision did not work anymore, and the scientist **decided to measure execution time of system commands in boards from “China” and “Canada”**. Below is the result “**without MBLH**”:



```
------------------СТАРТ ТЕСТА-------------------
--------ИМЯ ТЕСТА-----------СРЕДНЕ ВЗВЕШЕННОЕ ВРЕМЯ-------РЕЗУЛЬТАТ АНАЛИЗА---
Когерентность таймеров             299          Гипервизор не обнаружен
Команды MOV (CR0..Cr3)             235          Гипервизор не обнаружен
Команды MOV (DR0..Dr7)             237          Гипервизор не обнаружен
Команды RDMSR, WRMSR               235          Гипервизор не обнаружен
Очистка буферов TLB                234          Гипервизор не обнаружен
Команды виртуализации              233          Гипервизор не обнаружен
Команды Ввода/Вывода               233          Гипервизор не обнаружен
Монотонность доступа к ОП          259          Гипервизор не обнаружен
Монотонность доступа к БИОС        236          Гипервизор не обнаружен
Доступ в APIC                      235          Гипервизор не обнаружен
Доступ в NIC RAM                   234          Гипервизор не обнаружен
Выполнение прерываний              236          Гипервизор не обнаружен
-----------------Конец ТЕСТА--------------------
```

Case #1 – Malicious BIOS Loaded Hypervisor (11)



```
------------------СТАРТ ТЕСТА-------------------
--------ИМЯ ТЕСТА-----------СРЕДНЕ ВЗВЕШЕННОЕ ВРЕМЯ-------РЕЗУЛЬТАТ АНАЛИЗА---
Когерентность таймеров                8273          Присутствие Гипервизора
Команды MOV (CR0..Cr3)                14023         Присутствие Гипервизора
Команды MOV (DR0..Dr7)                16191         Присутствие Гипервизора
Команды RDMSR, WRMSR                  15831         Присутствие Гипервизора
Очистка буферов TLB                   19046         Присутствие Гипервизора
Команды виртуализации                 14502         Присутствие Гипервизора
Команды Ввода/Вывода                  16593         Присутствие Гипервизора
Монотонность доступа к ОП             14436         Присутствие Гипервизора
Монотонность доступа к БИОС           16787         Присутствие Гипервизора
Доступ в APIC                         17898         Присутствие Гипервизора
Доступ в NIC RAM                      13443         Присутствие Гипервизора
Выполнение прерываний                 14055         Присутствие Гипервизора
```

Case #1 – Malicious BIOS Loaded Hypervisor (11)

Expressing concerns and presenting the case to various parties;

1. **Intel representative in Russia** – no response
2. **FSB (Federal Security Services**, former KGB) “Center for Information Protection” (CIP) mid-level executives and specialists – misunderstanding and almost no interest expressed.
3. **FSB CIP second meeting** – senior level executives and specialists - The meeting turned into a lecture. … and finally answered many questions. At the end of the meeting they thanked us, and said that the issue should be investigated in the framework of special research.” Finally – the case is dismissed.
4. **To prove the case** “…… I should write such a "add-on" myself. **I would not be able to put the "add-on" in the flash memory of the BMC, but can load the code in the main BIOS.** … My Last Judgment Day weapon **will utilize the "add-on" to kill the computer system by an external command** … actually perform single function task of wiping out BIOS flash chip when receiving a command to destroy the system.” **Finally he managed to show the demo to GasProm (state oil and gas enterprise0) security team**, but with the same success as to others. ***Dismissed.***

Case #1 – Malicious BIOS Loaded Hypervisor (12) Conclusion

1. **The post describes unique experience of identifying a malicious hypervisor, which is loaded before OS bootup** and right after system BIOS hardware initialization.
2. The post definitely raises some questions, because important technical information concerning experiments is missed. However, considering the post contents, **we give it 90% of chance being reality while leaving 10% to the benefit of doubt.**
3. The author identified, while missing details, **that the malicious hypervisor is embedded in BMC BIOS.** Last years' research on modern exploits of IPMI and its weaknesses completely correlates with that.
4. The author describes his experiments and the **development and improvement of MBLH up to the level when it cannot be identified** even by running yet another virtualization software on the same level.
5. **Experiments with commands' execution time to identify the existence of MBLH are important but lacking details**, which would lower our suspicion level.
6. The author's ordeal while trying to convince authorities and technical professionals is very typical and well known to security professionals.
7. While **the author believes that there is "Chinese Hand"** in altering BMC BIOS, **we have some doubts, because the label "Assembled in China" does not mean anything,** like the label "Assembled in Canada" where we did not find Intel facility related to manufacturing and assembly process. ***By details of this case, MBLH might be developed by Kraftway or FSB.***

Case #2 – the US – Is where the Idea of Malicious Hypervisor Came from?

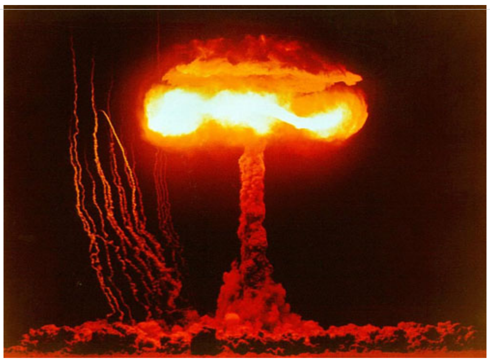
Not too much known research with prominent sponsors:
Joined team from University of Michigan and Microsoft Research did sponsored research “SubVirt: Implementing Malware with Virtual Machines” which (quote) “was supported in part by **National Science Foundation** grants CCR-0098229 and CCR-0219085, by **ARDA** grant NBCHC030104, by **Intel Corporation**, and by **Microsoft**”.
What is ARDA? Would then ARDA be now DARPA? Yes, many thanks to Wikipedia, **ARDA = DARPA (Defense Advanced Research Projects Agency)**.
Therefore, a while before Russian scientist started fighting with a ghost embedded in BMC BIOS, the foundation for malicious hypervisor has been researched, and two proof-of-concepts implemented sometime in 2005 – 2006 years.
This research is very important to understand various aspects of now existing problem of likely existing invisible malware and perspectives to mitigate that.

Case #2 - The Purpose of VMBR Research

Quote: **“We evaluate a new type of malicious software that gains qualitatively more control over a system. This new type of malware, which we call a virtual-machine based rootkit (VMBR), installs a virtual-machine monitor underneath an existing operating system and hoists the original operating system into a virtual machine.** Virtual-machine based rootkits are hard to detect and remove because their state cannot be accessed by software running in the target system.”
VMBR is different from MBLH, because it is not embedded in BMC BIOS but loaded in RAM, and thus has different functionality. **Just the first step …**
Quote: “**Our project which we called SubVirt, shows how attackers can use virtual-machine technology to address the limitations of current malware and rootkits.**
…We show how attackers can install a virtual-machine monitor (VMM) (comment: hypervisor) underneath an existing operating system and use that VMM to host arbitrary malicious software. The resulting malware, which we call a **virtual machine based rootkit (VMBR), exercises qualitatively more control than current malware, supports general purpose functionality, yet can completely hide all its state and activity.**”

Case #2 - Details of Ultimate Weapon Research

The research goals:
“**We demonstrate that a VMBR can be implemented on commodity hardware and can be used to implement a wide range of malicious services**.”
“We show that, once installed, a VMBR is difficult to detect or remove.”
“**We implement proof-of- concept VMBRs on two platforms** (Linux/VMware and Windows/VirtualPC) **and write malicious services** such as a keystroke sniffer, a phishing web server, a tool that searches a user’s file system for sensitive data, and a detection countermeasure which defeats a common VMM detection technique.”
“**Finally, we discuss how to detect and defend against the threat posed by VMBRs** and ***we implement a defense strategy suitable for protecting systems against this threat***.”

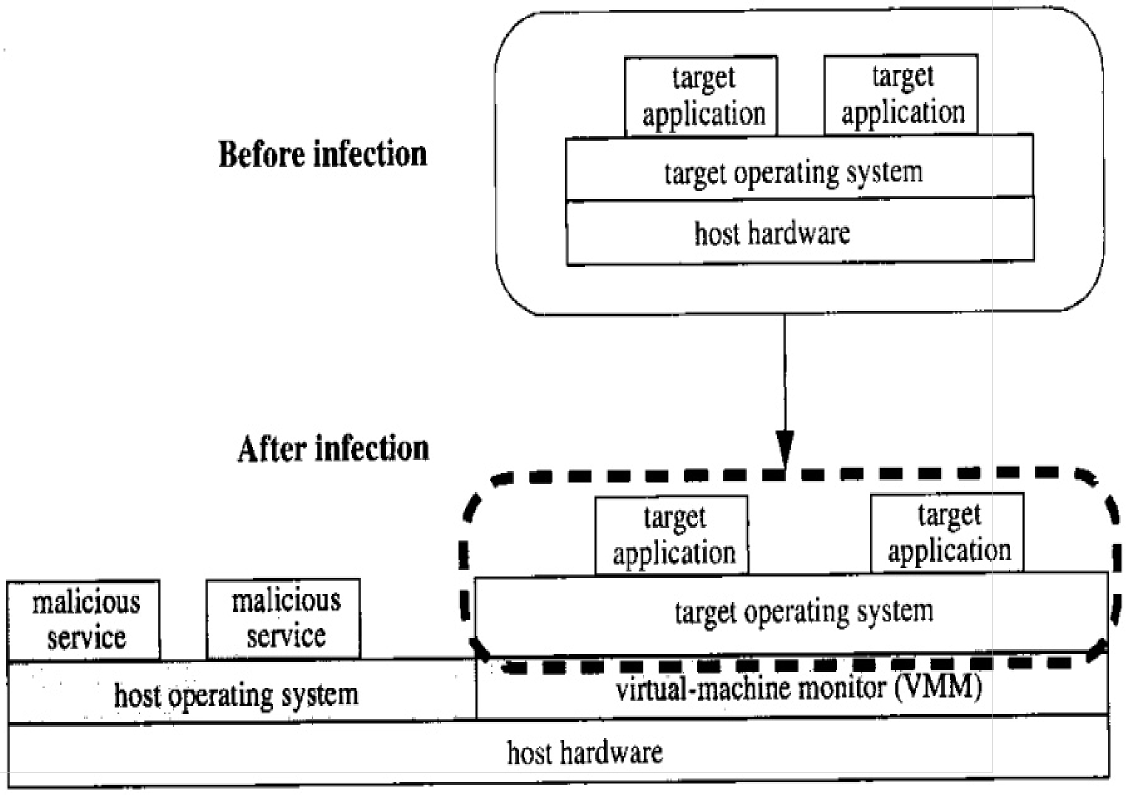


Case #2 – How to Control Guest OS

In “normal” virtual machine implementation there is no need for host OS to completely control guest OS. However, VMBR requires that:
“ … **One problem faced by VM services is the difficulty in understanding the states and events inside the guest they are serving**; **VM services operate at a different level of abstraction from guest software.** Software running outside of a virtual machine views low level virtual-machine state such as disk blocks, network packets, and memory. Software inside the virtual machine interprets this state as high-level abstractions such as files, TCP connections, and variables. **This gap between the VMM’s view of data/events and guest software’s view of data/events is called the semantic gap**.”
“Virtual-machine introspection (VMI) describes a family of techniques that enables a VM service to understand and modify states and events within the guest. VMI translates variables and guest memory addresses by reading the guest OS and applications’ symbol tables and page tables. VMI uses hardware or software breakpoints to enable a VM service to gain control at specific instruction addresses. “

Case #2 – Design and Implementation Notes (1)

VMBR installation process (quote):
“In the overall structure of a VMBR, a VMBR runs beneath the existing (target) operating system and its applications. To accomplish this, a **VMBR must insert itself beneath the target operating system and run the target OS as a guest.** To insert itself beneath an existing system, a **VMBR must manipulate the system boot sequence to ensure that the VMBR loads before the target operating system and applications**. After the VMBR loads, it boots the target OS using the VMM. As a result, the target OS runs normally, but the VMBR sits silently beneath it.” ***This process is different from MBLH where the hypervisor is loaded from BMC and then boots the OS***.
Malicious services (quote):
“**Using our proof-of-concept VMBRs, we developed four malicious services that represent a range of services** a writer of malicious software may want to deploy. We implemented a phishing web server, a keystroke logger, a service that scans the target file system looking for sensitive files, and a defense countermeasure that defeats a current virtual-machine detector. To develop these services, we use the host OS as our attack OS (below):


Before infection
target application
target application
target operating system
host hardware
After infection
target application
target application
malicious service
malicious service
target operating system
host operating system
virtual-machine monitor (VMM)
host hardware

Case #2 – Design and Implementation Notes (3)

Malicious services:
The authors of this research definitely went much further than we originally discussed as malicious hypervisor and MBLH. They also introduced exploits on the top of malicious hypervisor. That is actually going to the very end of a system compromise.
Maintaining control – defending VMBR:
Quote: " …**The only time the VMBR loses control of the system is in the period of time after the system powers up until the VMBR starts**. if the BIOS boots a program on an alternative medium, that program can access the VMBR's state."
There are two measures to regain the control:
- **Handling reboots by restarting the virtual hardware rather than resetting the underlying physical hardware** thus creating an illusion of reboot.
- **Emulation of system shutdown by using ACPI (Advanced Configuration and Power Interface) sleep states to switch hardware in lower power mode**, **which look s like a shutdown while memory state is intact**. The most targeted systems like ***servers are really rear shutdown***.

Case #2 – Design and Implementation Notes (4)

Evaluation of the implementation:
Quote: “This section evaluates the impact of a VMBR on a system. We valuate the disk space used by a VMBR, the time to install a VMBR, the effect of a VMBR on the time to boot the target OS, the impact of a VMBR as viewed by a user, and the effect of the memory space used by a VMBR.”
Comment:
The section contains interesting technical material, and could be helpful in further research of MBLH case. **However, the VMBR research has been done on systems not yet supporting hardware virtualization, so dependences and numbers will be completely different for modern systems starting from 2007 year like in MBLH case.**

Case #2 - Defending against VMBR (1)

This is possibly the most interesting part of this research – **how the creators of ultimate weapon suggest protecting against it.** Quote: “In this section, we explore techniques that can be used to detect the presence of a VMBR. VMBRs are fundamentally more difficult to detect than traditional malware because they virtualize the state seen by the target system and because an ideal VMBR modifies no state inside the target system. “
There are two methods:
Security software *below* the VMBR:
1. **To run a “detector” software below VMBR** – **by the VMBR concept, only system BIOS is below, so this recommendation is not concrete.**
2. **Secure boot prevents loading drivers and other software modules not having a “signature”, which compared with “platform key” in a firmware**. However, that is the function of boot loader to compare the key and signatures, but **one of the main functions of VMBR is the loader alteration**. **Secure boot is actually above VMBR, not below**.
3. **Other methods are secure hardware, the boot from save media and secure virtual machine monitor.** All of them will try to boot by other than affected hard drive. However, **we’ve seen a malware which blocks any boot but the hard drive** and does not matter what is configured in BIOS.

Case #2 - Defending against VMBR (2)

Secure Boot – continue:
“Using a secure VMM, we implemented an enhanced version of secure boot which can prevent VMBR installations. The goal of our secure boot system is to provide attestation for existing boot components, such as the disk’s master boot record, the file system’s boot sector, and the OS’s boot loader and also to allow legitimate updates of these components. All attempted updates of these components are verified (by checking the cryptographic signature) before they are allowed to complete.”
Our opinion:
We see some inconsistency in this concept. When such “secure VMM is installed? If before VMBR, that each computer system requires such component. If after VMBR is installed first,it controls any other installations, thus can prevent creating VMM on its own level. **We, in general, disagree with that anything could be installed below VMBR, because it is installed right after BIOS hardware initialization and initial tests, and thus can control any other installation.**

Case #2 - Defending against VMBR (3)

Software above VMBR - many obstacles:
1. **General obstacle to run security software above VMBR is, as discussed, its position above** and thus it is completely controlled by VMBR. **However, if VMBR utilizes all computer hardware resources, then (quote) "… timing differences can be noticed by software running in the virtual machine by comparing the running time of benchmarks against wall-clock time**".
The overhead in CPU, memory, hard drive could be measured, ***and then may be altered by VMBR as well. However, external clock will deliver unaltered time.***
We would like to note here that ***such measurements require two computer states – before and after VMBR installation, or two identical systems to compare, or somehow disabling VMBR***. We will discuss our experiments and conditions of "overhead" testing below.
2. Another option is utilization of HIDS system which would test installed software , for instance, I/O drivers for changes which required for virtualization.

Case #2 - Conclusion

Case #2 goals:
- The design and implementation of Virtual Machine Based Rootkit, Implementation of malware utilizing VMBR features, Evaluation of VMBR, Implementation on two platforms, Detection and defense against VMBR.
The research and the paper representing it are perfectly done, and convincing that all practical implementations worked according to the research theory and goals.
Our opinion:
Authors finally tried to downgrade potential threat of VMBR-like systems and leverage a possibility to overcome it.
We do not share authors' pessimism over VMBR deadly capabilities and optimism over detection and defeat. "Russian Ghost" Case #1 perfectly correlates with VMBR concept, and goes further by exploiting BMC capability to run malicious hypervisor on the lowest level possible. We believe that such hypervisor could even run before system BIOS thus virtualizing it and its functions.
By our opinion, there are two main results – the proof-of-concept and the code of VBMR. ***The research has been done before IEEE symposium in May, 2006. Considering that the research described in "Russian Ghost"" Case #1 began in 2007 or later, there is complete time correlation. The concept and the code could be used to move VMBR to the next level – BMC BIOS based MBLH. The threat level thus has significantly increased*.**

Our Case – Global or Local Threat and Possible Implementation

The research in the Case #2 lowered our doubts that the Case #1 is a myth. ***We see definite correlation and the development of initial idea of VMBR. However, even if Case #1 were complete myth, sooner or later the idea of VMBR would find its supporters and implementers in the form of MBLH.***

In the following paragraph we would like to identify the place and the vector of VMBR/MBLH threat. We will do our analysis bases on two already discussed cases first, and then will review our conclusion considering the Case #3, which follows.

Global or local threat? That is may be the most important question. It depends on how easy is to deploy and to distribute the threat.

Our case – VMBR and MBLH Deployment

Case #2 (VMBR) does not consider any particular deployment rather than RAM, while Case #1 (MBLH) pinpoints BMC BIOS. BMC BIOS has its own firmware file, which is to be deployed together with others.
The entire BOIS file is signed by Intel QA, and downloaded from authorized resource. Thus, **it is very unlikely that official Intel BIOS file contains an alteration, and more likely it happened after motherboard manufacturing, or server assembly process**.
We think that alteration process with associated logistics is very manual and is done for particular targets. We do not see a possibility of mass production of altered board to deploy around the globe.
So, it is very unlikely that a maniac hijacked Intel facility and spreads altered software with the purpose of pressing a button "Die All". However, **the access to data sources is very valuable in some cases.**

Our deployment conclusion: it is more feasible to alter some motherboards or servers targeting highly valuable data resources. Thus we dismiss the global option and incline to consider local MBLH deployments

Our case – VMBR and MBLH Distribution

Distribution:
Computing resources with MBLH will be physically shipped to targets as the first part of infiltration process. Then, utilizing MBLH resources and local network vulnerabilities other servers could be hijacked and converted to silent and invisible bots. Such process of searching and converting local targets does not require external resources excepting sites for downloading altered BIOS versions.
Our Case conclusion:
1. So far, based on what we know by two cases, **we do not expect worldwide distribution of MBLH infected computers because of high complexity of BMC BIOS alteration process and deployment**. Targeted “customers” may receive a shipment with altered BMC BIOS, and thus entire local network could be eventually hijacked.
2. **Concerning uploading VMBR in a computer system RAM, Case #2 research did not really consider options**. That could be local malicious activity of IT personnel uploading VMBR from a CD-ROM, etc. **In any case, local uploading is really limited as means of VMBR distribution.**

Case #3 – Widespread Distribution of Malicious Hypervisor via IPMI vulnerability

Now is the Case #3 - we return to Case #1 but with different perspective.
Michigan University research **“Illuminating the Security Issues Surrounding Lights-Out Server Management”** by Anthony J. Bonkoski, Russ Bielawski and J. Alex Halderman. Their research gave us new information applicable to already considered cases.
Quote: “**This paper examines the security implications of the Intelligent Platform Management Interface (IPMI), which is implemented on server motherboards using an embedded Baseboard Management Controller (BMC).** We consider the threats posed by an incorrectly implemented IPMI and present evidence that IPMI vulnerabilities may be widespread. “
“**We analyze a major OEM’s IPMI implementation and discover that it is riddled with textbook vulnerabilities, some of which would allow a remote attacker to gain root access to the BMC** and potentially take control of the host system.
Using data from Internet-wide scans, we find that there are at least 100,000 IPMI-enabled servers (across three large vendors) running on publicly accessible IP addresses, contrary to recommended best practice”.

Case #3 – Introduction and Related Topics

IPMI: Intelligent Platform Management Interface has been known since 1998, is independent from OS and functions even when computer system is down. **For yet unknown reason, up until 2012 – 2013 IPMI insecurity was not widely discussed, with the exception of our Case #1**, **when Russian scientist pointed out to BMC embedded malicious hypervisor.**
The research lists various known exploits, which could be installed in BMC. Quote: **"Malware residing on the BMC could be extremely difficult to detect, since it sits at an even lower architectural layer than a BIOS or VM-based rootkit (reference to VMBR)".**
It is considered as common sense security practice not to connect IPMI devices (i.e. its network controller) to public network. However (quote)"… we use data from Internet-wide surveys **to reveal public IP addresses of over 100,000 IPMI devices, including more than 40,000 systems that our results suggest are remotely exploitable."**
Quote:"…Instead of securing IPMI vendors do rather opposite: "**The vulnerabilities we find, along with others previously found …, suggest that some IPMI manufacturers are systematically failing to properly secure these devices."**

Case #3 – IPMI Security Risks

All known BMC OS use Linux OS which as any requires security updates. "IPMI malware carries similar threats (like BIOS) and is likely easier to develop, since many BMCs run a standard operating system. BMC malware would also likely be easier to install remotely, due to IPMI's substantial network-facing attack surface."

Attack surface: IPMI (Supermicro Inc. implementation) has six TCP and UDP ports for communication and management. **Even if recommended separate management network is used, there still is an opportunity to connect to BMC exploiting various issues**.

Authentication risks: Administrators tend to using default or the same password on multiple servers. And IPMI device may use insecure password storage. Thus, after compromising one system it is possible to get access to multiple.

Case #3 – Attack Scenarios

1. Subverting the host system- by exploiting remote management capabilities and thus accessing the most of server configuration
2. BMC spyware against host's OS– Quote: "If the attacker can install malware on the BMC, it would have a powerful vantage point for spying on the system and its administrator."
3. Persistent attack from BMC – Quote: "As the BMC operates independently from the host's operating system and CPU, it provides an ideal **hiding place for a stealthy, highly persistent rootkit….A BMC rootkit would survive reinstallation of the host's OS, or even complete replacement of the host's storage devices. … could even be designed to survive BMC firmware updates by dynamically patching the new firmware**."
4. Attacking BMC from the host system – an attacker could re-flash the BMC's firmware from compromised host's OS.
5. IPMI botnets – Quote: "If widely used IPMI devices can be compromised remotely, they can be leveraged to create large networks of bots. … **the system operator is unable to run normal malware detection and removal tools within the BMC.**"
We see that 4 out of 5 scenarios tightly correlate with the Case #1 claim that malicious hypervisor had been embedded in BMC firmware.

Case #3 – Analysis, Attacks and Network measurements

Quote: “To explore the potential for BMC compromise, **we analyzed an IPMI implementation shipped by one large server manufacturer, Supermicro**. … **We ultimately discovered a range of vulnerabilities, and we developed two proof-of-concept exploits to demonstrate some of the most critical problems**.”
Vulnerabilities and associated attacks:
Insecure input validation, Shell injection vulnerabilities, Buffer overflow vulnerabilities and Buffer overflow exploit.
(Quote) **“ … These vulnerable firmware images apply to 135 Supermicro product models. The problems may also affect IPMI devices from other manufacturers that are based on similar ATEN firmware.”**
Network measurements:
Quote: “… we used data from an Internet-wide network survey conducted in May 2013”. **While the search was limited to certificates’ analysis, the authors were able to identify over 105,000 servers of major manufacturers (Dell, Supermicro and HP) connected to Internet, and likely having vulnerabilities**.

Case #3 - The research conclusion

Quotes:
"… **Since BMCs operate independently of the host system and CPU, cleverly written malware running there *could potentially reside undetected indefinitely*. Unfortunately, due to the closed nature of BMC firmware, server operators have few avenues to defend themselves without vendor assistance."**
" … We uncovered a wide range of vulnerabilities and demonstrated two working attacks that allowed us to gain root shell access. These problems pose an immediate threat to many systems in the field; we found over 40,000 devices similar to the one we analyzed visible on public IP addresses."
"…**In the long run, securing remote management systems calls for a defense-in depth approach. Vendors need to apply careful security engineering practices, minimize attack surfaces, and help users ensure that their systems are appropriately locked down and isolated from public networks**."

Our Case #3 conclusion

1. The research radically changes our preliminary conclusion given in p.4.2. Now, combining all three cases, **we see global threat by VMBR/MBLH distribution via IPMI vulnerability**, which is the combination of very low security of IPMI/BMC implementations and human factor – simply security ignorance of IT personnel. Numbers shown in the Case #3 research are not final, and may turn in to millions of ready for exploits servers.
2. **We do not think that the situation will improve in near future, because of costs of IPMI/BMC securing and required time.** Neither we are optimistic over the "human factor" and IT attitude.
3. **We see that modern trend toward "cloud services" may affect overall information security.** Users of in most cases do not understand that "clouds" are simply monstrous datacenters with thousands of computers, and all of them serve users over Internet. **Having thousands of opened Internet connections, even with all security measures, represent high risk of one exploited and then all infected over shared vulnerable management network**. Needless to say that such centers are very likely to have management systems Internet connected as well.

Our Ten Cents to How to Identify MBLH Presence

We share the Case #3 opinion that identification from "below" simply does not exist as VMBR/MBLH is always on the lowest level, for instance embedded in BMC/IPMI.
Catching by 100% utilization and in "statistical net":
1. **Instead of measuring commands' execution time (Case #1), use 100% utilization** (Case #2) and watch on software execution time.
2. **We need to get two testing options – with or without MBLH. It is possible only if computer system BIOS supports turning hardware virtualization on or off.** Not all chip and server vendors support that. ***Our Dell T105 AMD Opteron based servers have virtualization always up – no BIOS settings for***.
3. **However, hardware virtualization will work if MBLH is not designed to reside below system BIOS**. In this case it will be able to alter BIOS function of turning virtualization off.
4**. We used supporting virtualization and BIOS settings Lenovo notebook** with Linux CentOS 6.6 OS installed on.
5. **This computer system BIOS has two virtualization related options** - Intel Virtualization Technology (Disable/Enable) and Intel VT-d Feature (virtualization technology support for direct I/O, Disable/Enable).

Our Ten Cents – Measurements and Statistical Net

No Virt.				**Virt.**			
Test #	Time	Dev	(Dev)2	Test #	Time	Dev	(Dev)2
31	19:54	1	1	41	19:57	5	25
32	19:49	-4	16	42	19:45	-7	49
33	19:55	2	4	43	19:50	-2	4
34	19:59	6	36	44	19:48	-4	16
35	19:46	-7	49	45	19:54	2	4
36	19:43	-10	100	46	19:40	-12	144
37	19:58	5	25	47	20:03	11	121
38	19:55	2	4	48	19:57	5	25
39	19:50	-3	9	49	19:59	7	49
40	20:00	7	49	50	19:49	-3	9
Mean:	19:53	Sum:	293	Mean:	19:52	Sum:	446
SD			5.4			SD	6.7

Our Ten Cents – Measurements and Statistical Net (2)

Our testing software “dowork” (it is available for public) does **floating point calculation driving CPU to 100% utilization**. Each calculation cycle takes a few seconds. **We used 400 cycles to get statistically more stable results**.
However, various OS processes created certain deviation, thus **we used 10 tests to improve statistics in both cases when two virtualization options are disabled and when enabled**.
Our results presented in the following table where first four columns are for the test with disabled virtualization and next four – with virtualization; Dev is deviation, SD – standard deviation, time is “minutes:seconds” format.
We see that for both tests mean values are very close – 19:53 and 19:52.
Standard deviation in the case of enabled virtualization is higher, but **statistically both results are very close – SD is 5.4 and 6.7.**

We did not find - statistically – the presence of a malicious hypervisor.

We think that such statistical testing approach on long running programs is the only one way to overcome live OS variations in execution time and is likely to identify the presence of a malicious hypervisor.

Our Ten cents – External MBLH Catching - SIEM

There is one more potential opportunity to catch VMBR/MBLH. Modern Security Information and Event Management System (a SIEM is not a luxury anymore, but security necessity) may help to identify short and infrequent connections of a malicious hypervisor via management or system network interface to internal or external hosts. Basically, we are facing to find a needle in a hay stock, but this a SIEM is for – logging random and seldom events across entire network and systems and finding suspicious connections. We did not implement such hunt simply because we do not have servers with hardware virtualization support disabling functionality in system BIOS, but we encourage the audience not to ignore such opportunity if you have a few servers to experiment with and a SIEM in place.

Our research Conclusion

1. Case #1 – Russian mythical post. It is the only one but yet circumstantial evidence of successful attempt to embed a malicious hypervisor in BMC BIOS software. We consider it as “reality”, but give 10% to the benefit of doubt. **Its reference to BMC as the platform to run a malicious hypervisor completely correlates with two other cases.** The post author observed for long time continuing improvement of MBLH software until it became completely invisible to the second hypervisor. We used proposed test of execution in our attempt to find MBLH in our Lenovo notebook. **There is neither real evidence nor logic to suspect that MBLH has been embedded during manufacturing process in China.** May be later … at any place in the world.
2. Case #2 – US – Michigan University research ended approximately in 2006. Virtual Machine Based Rootkit (VMBR) is the fundamental research proving that malicious hypervisor can be developed and work on lower that OS level, and can move successfully OS one level up and replace it by itself occupying the lowest system level. **We think that this concept and may be the code itself have been used in the Case #1. We believe that the code has been almost publicly available and finally has been developed in Malicious BIOS Loaded Hypervisor (MBLH).**
The research authors were trying to downplay the danger of the solution and proposed two possible ways to identify and protect against it. **However, the method “from below” simply does not exist, and as case #1 showed, the hiding capability of VMBR is extraordinary.**

Our research Conclusion (2)

3. Based on Case #1 and Case #2 we do not expect worldwide distribution of MBLH infected computing resources because of high complexity of BMC BIOS alteration process. Targeted “customers” may receive a shipment with altered BMC BIOS, and thus entire local network could be eventually hijacked utilizing 0-day attacks and updating original BIOS to altered version.
4. Case #3 - The research on IPMI vulnerability, which has been done at Michigan University and published in 2013, **has changed the VMBR/MBLH threat landscape significantly. The research showed that production servers have numerous vulnerabilities, which make BMC and then entire system easy to compromise target. Negligence and ignorance of IT personnel managing server platform created situation when hundreds of thousands servers around the globe are Internet connected and thus can be easy exploited**. We now see that the most likely scenario for VMBR/MBLH distribution is over vulnerable BMC implementations either Internet or LAN connected.
5. Cloud - The most dangerous situation, as we see that, is at “cloud” services datacenters, which have hundreds of servers working with users over Internet; it is possible that these servers’ management system also connected to public network.
6. Even without Case #1**, it would be enough two MU research projects to get us to the same conclusion about worldwide threat of VMBR embedded in BMC and exploiting IPMI.**

Pure Summary

- Russian Ghost/VMBR/MBLH is dangerous as can infiltrate in millions of servers worldwide
- The hunting season is opened but be patient – we have only two perspective tools so far
- In theory, we cannot identify it, but we still have a chance …
- There is no protection against it – put your server is a dumpster – special thanks to IPMI
- ***NO security standard calls for server management (IPMI) protection***

References

1. Chinese Add-ons: True Stories of virtualization, information security and computer spying; post on http://xakep.ru/articles/58104/ 12/26/2011. Translated from Russian, Copyright © DeepSec, GmbH and Rubos, Inc., 2014.
2. Intelligent Platform Management Interface. Wikipedia, see http://en.wikipedia.org/wiki/Intelligent_Platform_Management_Interface
3. SubVirt: Implementing malware with virtual machines. Samuel T. King, Peter M. Chen (University of Michigan); Yi-Min Wang, Chad Verbowski, Helen J. Wang, Jacob D. Lorch (Microsoft Research); IEEE Symposium on Security and Privacy, Berkley/Oakland, CA, USA, 21-24 May, 2006.
4. Illuminating the Security Issues Surrounding Lights-Out Server Management by Anthony J. Bonkoski, Russ Bielawski, J. Alex Halderman; Michigan University. 7Th USENIX Workshop on Offensive Technologies, August 13, 2013, Washington, DC. https://www.usenix.org/conference/woot13/workshop-program/presentation/bonkoski
5. Unified Extensible Firmware Interface, Wikipedia; http://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface

Thank you!

All questions will be answered:

- mikhailutin@hotmail.com

or

- mutin@rubos.com

- This presentation will be available on DeepSec site