# ОБЕСПЕЧНИЕ БЕЗОПАСНОСТИ BIOS С ПОМОЩЬЮ ГИПЕРВИЗОРА

Дементьев З.В., студент кафедры безопасных информационных технологий ИТМО, zd3men@gmail.com

Ханов А.Р., тьютор кафедры безопасных информационных технологий ИТМО, awengar@gmail.com

## Аннотация

BIOS – базовая система ввода-вывода. BIOS является неотъемлемой частью компьютера, так как именно он подготавливает аппаратную часть компьютера к работе. Однако, сегодня данная технология является гораздо больше чем просто код подготавливающий систему к запуску. Поэтому BIOS становится выгодной целью для атак.

В данной работе приведен краткий обзор атак на BIOS, существующих механизмов защит, а также предложен механизм защиты, основанный на технологии аппаратной виртуализации.

## Введение

Во многих областях своей деятельности человечество движется по пути проб и ошибок. Сфера информационной безопасности, и в частности сфера безопасности прошивок компьютеров, не стала исключением. Бурный рост компьютерных вирусов, а также научные исследования показали, что к безопасности системного программного обеспечения нужно относится особенно серьезно. Ведь фактически при компрометации BIOS, компрометируется вся система целиком. Рассмотрим далее, чем же грозят атаки на BIOS и как им противодействовать.

## Векторы атак

1. Хранилище прошивки

Вредоносный код получает право на запись в SPI-флеш. Самое простое, что он может сделать - это испортить содержимое памяти, обеспечив таким образом DOS. Другой вариант – заразить BIOS. Это даст возможность вредоносному коду влиять на ход загрузки операционной системы, выключать механизмы защиты, оставаться независимым от операционной системы. Также такой руткит будет довольно сложно обнаруживать, так как он может не оставлять следов на диске и даже в

оперативной памяти (например, при заражении SMM модуля).

2. SMM (System Management Mod)

Данный вектор атаки является очень перспективным. SMM является более привилегированным режимом исполнения чем режим ядра операционной системы. Переход в данный режим осуществляется посредством механизма прерываний, как программных, так и аппаратных. Если вредоносный код каким-либо образом сможет выполнить код из данного режима, то это повысит его шансы на осуществление первой атаки. Если произойдет заражение SMM модуля, то вредоносный код будет недетектируем, так SMRAM недоступна ни для ядра операционной системы, ни для гипервизоров.

3. S3BootScript

Основная цель данной технологии заключается в восстановлении состояния компьютера после выхода из сна, в том числе в восстановлении механизмов защиты. Если вредоносный код сможет определенным образом модифицировать S3BootScript, то он получит возможность провести предыдущие атаки еще до того, как включатся механизмы защиты.

4. NVRAM

Получив доступ ко всем настройкам UEFI, вредоносный код может отключать механизмы защиты, что упростит выполнение предыдущих атак. Также ошибки в реализации управления переменными в NVRAM могут привести к DOS атаке, когда после хаотичного или не очень изменения настроек машина вообще не сможет загрузится.

## Механизм защиты BIOS_CNTL

BIOS_CNTL (Bios Control Register) — специальный регистр чипсета, который реализует сразу два механизма защиты от атак на хранилище прошивки [1]. Первый механизм заключается в блокировке чипсетом команд на запись. Реализуется двумя специальными битами: первый бит (BIOSWE) отвечает за именно за возможность записи, второй бит (BLE) будучи установленным блокирует возможность изменить значение BIOSWE. Таким образом данный механизм защиты мог бы гарантировать, что никакой код из ring 0 не сможет изменить содержимое микросхемы с BIOS. Однако в 2015 году на конференции “31C3” было продемонстрированно, что данная технология уязвима к атаке “race conditions” [2].

Второй механизм заключается в блокировании команд на запись, если эта запись происходит не из SMM. За обновление прошивки отвечает

специальный SMI-обработчик, который проверяет цифровую подпись прошивки и в случае успеха выполняет запись. За использование этого механизма отвечает бит SMM_BWP (после установки бит нельзя сбросить). Аналогичная технология от AMD называется SpiRomProtect.

К сожалению, нередко эти защиты настроены неправильно или отключены вовсе. Поэтому применим технологии аппаратной виртуализации для закрытия уязвимости в виде ненастроенного механизма защиты BIOS_CNTL.

## Реализация защиты с помощью гипервизора

### *Требования*

Определим основные требования для гипервизора:

- Инициализировать BIOS_CNTL при загрузке гипервизора, выставив бит BWE в 0, BLE в 1, SMM_BWP в 1;
- Перехватывать попытки записать 1 в бит BWE (отвечает за возможность записи в SPI-флеш). Как уже упоминалось ранее существует эксплоит, который обходит эту защиту. Данная мера нейтрализует этот эксплоит.
- Восстанавливать значение BIOS_CNTL при выходе из спящего режима, поскольку его значения сбрасываются при переходе в этот режим.

Для реализации поставленной цели был выбран гипервизор Bitvisor с открытым исходным кодом. Его отличительной особенностью является небольшой объем исходного кода и архитектура [3]. Также его преимуществом является то что контролируемая операционная система не нуждается во внесении каких-либо изменений в свой код. Однако в силу своей архитектуры Bitvisor теряет возможность выполнять роль монитора множества виртуальных машин.

### *Реализация*

Доступ к регистру BIOS_CNTL осуществляется через конфигурационное пространство PCI с адресом 0:31:0 (bus:device:function) по смещению 0xDC [1]. Далее приведены основные участки кода, ответственные за работу защиты, которые внесены в файл pci_core.c.

Листинг 1.

```
//Используемые константы
#define BWE_BIT 0x00000001
```

```
#define BLE_BIT 0x00000002
#define SMM_BWP_BIT 0x00000020
#define BIOS_CNTL_ADDR 0x8000F8DC
// Инициализация регистра BIOS_CNTL
void vt_init_bios_cntl() {
   u32 data = -1;
   spinlock_lock(&pci_config_lock);
   out32(PCI_CONFIG_ADDR_PORT,
BIOS_CNTL_ADDR);
   spinlock_unlock(&pci_config_lock);
   in32(PCI_CONFIG_DATA_PORT, &data);
   data |= BLE_BIT;
   data |= SMM_BWP_BIT;
   data &= ~BWE_BIT;
   out32(PCI_CONFIG_DATA_PORT, data);
}
// Код фильтра записи по адресу
BIOS_CNTL
if (io.dir == CORE_IO_DIR_OUT) {
   if (current_config_addr.value ==
BIOS_CNTL_ADDR)
      data->dword &= ~BWE_BIT;
   pci_handle_default_config_write (dev,
io.size, offset, data);
}
```

Для использования предложенного метода защиты на UEFI системе необходимо:

- скомпилировать модифицированный BitVisor и загрузчик (идет в комплекте с исходным кодом BitVisor);
- Установить на флеш-диск менеджер загрузчиков (в нашем случае использовался rEFInd) и скопировать модифицированный гипервизор и загрузчик гипервизора;
- при загрузке выбрать в меню загрузчик BitVisor. После того как он загрузится управление вернется менеджеру загрузчиков;
- Выбрать загружаемую операционную систему. Теперь она работает в виртуализированном окружении.

Использование BitVisor на Legacy системе, а также установка на жесткий диск описана в документации BitVisor [4]. После активации защиты, в ее работоспособности можно убедиться с помощью модуля common.bios_wp из фреймворка CHIPSEC [5].

## Заключение

В работе были рассмотрены основные угрозы для такого системного программного обеспечения как BIOS, рассмотрен один из существующих механизмов защиты. Также было показано, что технологии аппаратной виртуализации могут стать дополнительным механизмом, который дополнит и исправит недостатки сущестующих.

## Литература


1. Intel® 8 Series/C220 Series Chipset Family Platform Controller Hub (PCH) Datasheet, May 2014, pp/ 396.
2. R. Wojtczuk, C. Kallenberg, официальный сайт компании Bromium, «Attacks on UEFI Security», URL: https://bromiumlabs.files.wordpress.com/2015/01/attacksonuefi_slides.pdf.
3. Takahiro Shinagawa, Hideki Eiraku, Kouichi Tanimoto, Kazumasa Omote, Shoichi Hasegawa, Takashi Horie, Manabu Hirano, Kenichi Kourai, Yoshihiro Oyama, Eiji Kawai, Kenji Kono, Shigeru Chiba, Yasushi Shinjo, and Kazuhiko Kato. 2009. BitVisor: a thin hypervisor for enforcing i/o device security. In Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments (VEE '09). ACM, New York, NY, USA, 121-130. DOI=http://dx.doi.org/10.1145/1508293.1508311
4. BitVisor 1.1 Reference Manual (v. 1.0 December 2010).
5. CHIPSEC manual (v. 1.2.5. November 2016)