УДК 004.056.5(045)

ЗАХИСТ ДАНИХ ШЛЯХОМ МОДИФІКАЦІЇ СИСТЕМНИХ РОЗДІЛІВ ЖОРСТКОГО ДИСКА

Павлов В. Г., канд. техн. наук, доц.

Національний авіаційний університет

kszi@ukr.net

Розглянуто методи захисту інформації на електронних носіях. Наведено переваги шифрування системних розділів «вінчестера» над шифруванням окремих файлів, каталогів і томів. Продемонстровано шляхи посилення захисту інформації на жорсткому диску за рахунок зміни вмісту Master Boot Record, зокрема, Partition Table.

Ключові слова: захист інформації, електронні носії, шифрування, системні розділи, Master Boot Record, Partition Table, переривання BIOS.

There have been considered the methods of information security on electronic media. There have been shown the advantages of the «winchester's» system partition encryption over the encryption of specified files, folders and volumes. There have been demonstrated the ways to make information security stronger due to the change of the Master Boot Record content, in particular, Partition Table.

Keywords: information security, electronic media, encryption, system partitions, Master Boot Record, Partition Table, BIOS interrupts.

Постановка проблеми

Зберігання інформації в комп'ютері пов'язане з ризиком її випадкового або навмисного розкриття, тому для її захисту застосовують різні методи і засоби. Найчастіше в ролі основного засобу, що перешкоджає вільному доступу до інформації в електронному вигляді, використовують шифрування, яке реалізується апаратним або програмним способом. Як об'єкти криптографічного захисту виступають окремі файли, групи файлів, каталоги і, нарешті, логічні диски. Результатом шифрування є файлові архіви, що знаходяться на електронному носієві. Не вдаю-чись до подробиць реалізації різних криптографічних алгоритмів, можна відзначити, що в цьому випадку необхідно також вирішувати завдання, пов'язані з гарантованим знищенням початкової незашифрованої інформації, щоб виключити її відновлення.

Річ у тому, що особливістю всіх використовуваних файлових систем (FAT, NTFS та ін.) є неможливість перезапису інформації в кластерах, що належать якому-небудь файлу, доки даний файл не буде помічений, як видалений. Тобто під час шифрування файла, його зашифрований екземпляр записується не замість початкового файла, а у вільні кластери електронного носія, і лише після завершення процесу шифрування незашифрований файл віддаляється, якщо це передбачено алгоритмом.

Таким чином, можливо відновлення видаленої незашифрованої інформації, чим з успіхом справляються такі програмні продукти, як «*File Recover*», «*Back2Life*», «*R-Studio*», «*GetDataBack for NTFS*» і ін.

Якщо не враховувати рекомендацій, пов'язаних з повним фізичним знищенням електронного носія, то унеможливити це можна тільки шляхом багатократного перезапису тих кластерів магнітного диска, де раніше зберігалася конфіденційна інформація. Наприклад, американський національний стандарт Міністерства оборони DOD 5220.22-M (E) припускає в перший прохід запис випадкових чисел, у другій — чисел, додаткових до записаних на попередньому проході, а в третій — випадкових чисел. У відомому алгоритмі Пітера Гутмана (Peter Gutmann), який вважається одним з найнадійніших, на місце знищуваних даних по черзі пишуться всі відомі комбінації розрядів (усього здійснюється 35 проходів). Схожі методи знищення інформації підтримують американські стандарти DOD 5220.22-M, Army AR380-19, NCSC-TG-025, Air Force 5020, NAVSO P-5239-26, HMG IS5, німецький VSITR, канадський OPS-II і російський ГОСТ Р50739-95 і ін. Таким чином, шифрування, що виконується всередині логічного диска, завжди зв'язане з ризиком неповного знищення початкової інформації, тому необхідний інший підхід до цієї проблеми.

Аналіз досліджень і публікацій

Тема захисту інформації, яка зберігається на електронних носіях, є достатньо актуальною, що підтверджується великою кількістю публікацій цього напряму. Умовно їх можна розділити на декілька груп по тому, на які аспекти роблять автори акцент у своїх роботах.

Першу, і найчисленнішу групу публікацій присвячено опису того або іншого програмного продукту і тому носять презентаційний характер.

У них, як правило, все зводиться до переліку можливостей, які реалізує програма, і режимів її роботи. Ця група публікацій найбільш корисна для тих, кому необхідно на практиці реалізувати захист інформації на електронних носіях за допомогою готової програми.

Друга група публікацій, присвячена основам криптографії, містить опис математичних перетворень, що реалізовуються певним алгоритмом шифрування. По суті, — це теорія, яка повинна знайти своє практичне застосування, тому дані публікації поза сумнівом корисні математикам-програмістам при написанні програм, що реалізують алгоритми криптозахисту.

І, нарешті, третя група публікацій, у яких разом з викладом теоретичних основ пропонуються шляхи реалізації даних методів на практиці. На думку автора, дані публікації найбільшою мірою відображають інтереси фахівців з комп'ютеризованих систем безпеки, оскільки містять практичні рекомендації і приклади, що демонструють отриманий результат.

Цілі

Мета роботи — аналіз методів захисту інформації, заснованих на зміні і перетворенні системних розділів жорсткого диска, і їх практична реалізація.

Принципи зберігання інформації на магнітних носіях

Для того, щоб розібратися в суті запропонованих шляхів вирішення даної проблеми, необхідно розглянути, яким чином здійснюється розміщення інформації на магнітних носіях. Як приклад візьмемо організаційну структуру накопичувача на жорстких магнітних дисках (НЖМД), яка в найбільш загальному вигляді характерна і для інших видів носіїв.

Як відомо, прийнято розділяти структуру зберігання інформації на носієві, як фізичному об'єкті і його логічну структуру. Якщо розглядати НЖМД як сукупність дисків, вкритих магнітним матеріалом, то для однозначного визначення місцезнаходження інформації на «вінчестері» використовують трикоординатну систему CHS, що отримала свою назву від трьох складових адреси інформаційного блока:

- номера циліндра або доріжки (*C-cylinder*);
- номера поверхні диска і відповідною їй магнітної головки читання/запису (*H-head*);
- номера сектора на доріжці (*S-sector*).

Ця структура і діапазон зміни величин *C*, *H* і *S* залежать від характеристик НЖМД як апаратного пристрою, які визначаються фірмою-виробником даних пристроїв і указуються, як правило, на корпусі «вінчестера».

Навпаки, логічна структура НЖМД визначається користувачем і формується їм самостійно перед початком експлуатації «вінчестера». Даний процес полягає у створенні в рамках загального інформаційного об'єму НЖМД одного або декількох логічних розділів (дисків, томів), на яких надалі і зберігатиметься інформація. Задля цього застосовують спеціальні програмні продукти, зокрема відому програму «*Partition Magic*».

Усі дані про створені розділи зводяться в спеціальну таблицю розділів — *Partition Table*, яка розміщується в службовому секторі **Master Boot Record** (MBR) [1]. Місцезнаходження MBR у системі координат CHS завжди постійно і відповідає значенням $C = 0$, $H = 0$ і $S = 1$.

Для кожного розділу в *Partition Table* відводиться 16-байте поле такого формату (рис. 1).

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Boot	HdS	SecS	CylS	Sys	HdF	SecF	CylF	SecBegL		SecBegH		CountSecL		CountSecH	

Рис. 1. Структура разделу *Partition Table*

У ньому містяться такі параметри розділу:

Boot — ознака завантажувального розділу;**HdS** — координата *H* початку розділу;**SecS** — координата *S* початку розділу;

CylS — координата *C* початку розділу;

Sys — тип файлової системи, що розгорнена усередині розділу;

HdF — координата *H* кінця розділу;

SecF — координата *S* кінця розділу;

CylF — координата *C* кінця розділу;

SecBegL — відносний номер початкового сектора (молодший байт);

SecBegH — відносний номер початкового сектора (старший байт);

CountSecL — кількість секторів у розділі (молодший байт);

CountSecH — кількість секторів у розділі (старший байт).

Partition Table має об'єм 64 байти, отже, вміщує чотири подібні поля, куди може бути поміщена інформація про чотири логічні диски, хоча на практиці використовуються не більше двох полів, оскільки логічні диски вкладаються один в одного на зразок «матрьошки».

Для кожного комп'ютера *Partition Table* його НЖМД є унікальною, тому становить особливу цінність з погляду на можливості доступу до інформації, що зберігається. Руйнування або спо-

творення *Partition Table* робить неможливим доступ до логічних дисків, а отже, і до тієї інформації, яка на них зберігається.

Послідовність завантаження програмного забезпечення комп'ютера.

Кожного разу, вмикая комп'ютер, слід повторювати процес завантаження і розгортання операційної системи. Оскільки файли операційної системи і прикладних програм зберігаються на логічних дисках НЖМД або інших зовнішніх пристроях, спочатку повинні бути забезпечений доступ до цих пристроїв і можливість читання з них даної інформації.

Ці функції виконуються за допомогою переривань — спеціальних програмних процедур, включених до складу BIOS (*Basic Input-Output Operating System*) — базової операційної системи вводу—виводу.

Територіально програма BIOS знаходиться в енергонезалежній пам'яті FLASH EEPROM, яка «прошивається» при виготовленні і встановлюється на материнську плату. Даний тип пам'яті розглядається як ROM (*Read Only Memory*) — пам'ять, що не підлягає зміні, оскільки зберігає вміст при вимкненні живлення комп'ютера. Для змінної частини програми BIOS використовується пам'ять CMOS (*Complementary-symmetry / Metal-Oxide Semiconductor*), технологія виготовлення якої дозволяє забезпечити довготривале живлення від портативного джерела (акумулятор або батарея) протягом тривалого терміну.

У разі ввімкнення живлення в комп'ютері програма BIOS завантажується першою. Спочатку з її допомогою виконується ряд тестів для перевірки працездатності базових апаратних компонентів комп'ютера, а потім, після благополучного завершення тестування, виконується пошук MBR на «вінчестері» або відповідних секторів на інших завантажувальних пристроях (дискета, *CD, Flash* та ін.)

Далі BIOS за допомогою свого завантажувача *Bootstrap Routine* читає з MBR програму завантажувача IPL1 (*Initial Program Loader*), розміщує її в пам'яті, починаючи з адреси 0000:7C00h і передає їй управління. У свою чергу IPL1 сканує *Partition Table*, знаходить активний розділ (логічний диск) і звертається до нього.

Подальший сценарій завантаження визначається вмістом системних розділів активного логічного диска, який залежить від файлової системи, що використовується на диску.

Структура деяких файлових систем

Як відомо, файловою системою називається спосіб організації зберігання файлів на логічному диску, який повинен забезпечувати:

- швидкий доступ до будь-якого файла або каталога на логічному диску;
- оперативний облік вільного місця на логічному диску і формування переліку незайнятих секторів або їх груп-кластерів;
- неможливість запису інформації одного файла в кластери, що належать іншому файлу, якщо цей інший файл не був видалений;
- виконання операцій копіювання, переміщення, перейменування і видалення над файлами або каталогами.

Файлова система на логічному диску формується при його форматуванні (ініціалізації).

У будь-якій файловій системі частина простору логічного диска відводиться для системних розділів, а інша — для зберігання файлів. Склад і зміст системних розділів визначається файловою системою.

Для файлової системи FAT формуються три системні розділи:

Boot record — займає один сектор на самому початку логічного диска. Містить завантажувач IPL2, а також інформацію про розташування і характеристики інших системних областей і параметри логічного диска: розмірі кластера, загальна кількість кластерів, призначених для зберігання інформації та ін.

Root — призначений для зберігання інформації про файли і каталоги, що знаходяться в кореневому каталозі логічного диска.

Для кожного файла або папці формується 32-байтне поле, в якому указуються всі їх основні характеристики: ім'я, розширення імені, розмір, час і дата останнього редагування файлу, атрибути файла або папки, номер кластера, з якого починається розміщення файла на логічному диску.

Якщо ім'я файлу довше, ніж 8 символів, то для його запису використовуються додаткові 32-байтні поля, які призначені для зберігання символів повного імені.

FAT (File Allocation Table) — призначений для зберігання інформації про номери всіх кластерів, у яких розміщується файл або каталог. Це дає можливість «зібрати» всі кластери файла навіть тоді, коли вони не розташовані підряд, і прочитати з них інформацію.

Важливість цього службового розділу підкреслюється тим, що он представлений у двох ідентичних копіях: FAT-1 і FAT-2.

Якщо логічний диск форматується для файлової системи NTFS, то на ньому формуються два системні розділи:

Boot Record — займає один сектор і його вміст аналогічно завантажувальному сектору FAT;

MFT (Master File Table) — містить інформацію про всі файли і теки з координатами їх розміщення на логічному диску.

Як випливає з наведеної вище інформації, і у файловій систем FAT, і у файловій системі NTFS серед системних розділів наявний **Boot Record**, що включає завантажувач IPL2.

Саме цей завантажувач приймає «естафету» від IPL1:

1. IPL1 проглядає записи розділів в **Partition Table** і аналізує перший байт кожного розділу. Якщо в ньому знаходиться код **80h**, то цей розділ розглядається як завантажувальний, якщо знайдений код **00h**, то завантаження з цього розділу неможливе. Якщо вміст байта відрізняється від вказаних кодів або код **80h** мають декілька розділів, то ця ситуація розглядається як помилкова, і завантажувач виводить повідомлення «***Invalid partition table***».

2. IPL1 зчитує перший сектор завантажувального логічного диска і поміщає його вміст у пам'ять, починаючи з адреси **0000:7c00**.

3. IPL1 перевіряє вміст двох останніх байтів цього сектора. Якщо код у них відрізняється від сигнатури **55AAh**, то перший сектор розпізнається як *Boot Record*, інакше завантаження переривається з виведенням повідомлення «***Missing operating system***».

4. IPL1 передає управління за адресою **0000:7c00**, тобто завантажувачу IPL2.

Головним завданням завантажувача IPL2 є пошук на завантажувальному логічному диску файлів операційної системи і передача ним управління для продовження завантаження. Оскільки вміст IPL2 визначається типом операційної системи, то завантажувач виконує пошук системних файлів з певною назвою.

Завершується робота IPL2 копіюванням цих файлів у пам'ять і передачею ним управління, після чого всі функції із завантаження комп'ютера виконуються операційною системою.

Послідовність завантаження комп'ютера у вигляді алгоритму показано на рис. 2

Захист інформації шляхом блокування завантаження комп'ютера

Цей підхід ґрунтується на тому, що доступ до інформації в комп'ютері можливий після завершення його завантаження і відкрито вміст логічних дисків.

Якщо ж завантаження штучно буде заблоковано на якій-небудь стадії, то, як передбачається, вміст логічних дисків буде неможливо прочитати.

Розглянемо основні прийоми блокування завантаження, спираючись на алгоритм, поданий на рис. 2. таким чином отримуємо такі варіанти:

1) блокування на рівні завантажувача BIOS;
2) блокування на рівні файлів і засобів операційної системи;
3) блокування на рівні завантажувача IPL2;
4) блокування на рівні завантажувача IPL1.

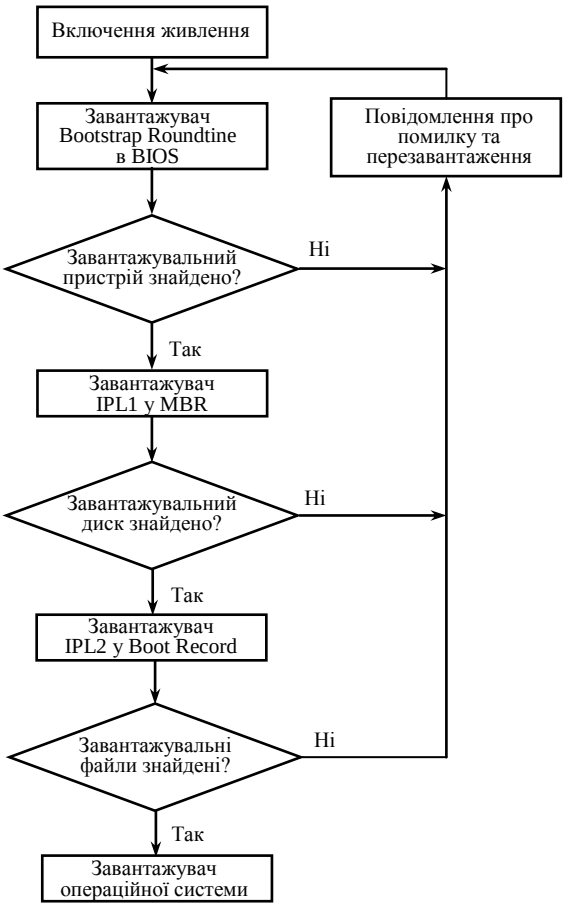

Рис. 2. Алгоритм завантаження комп'ютера

Проаналізуємо кожен варіант, відзначивши його переваги і недоліки.

1. Для блокування завантаження за допомогою BIOS не потрібно ніякого додаткового програмного забезпечення. Поставлену мету можна досягти засобами самого BIOS, виключивши, наприклад, НЖМД зі списку підключених зовнішніх пристроїв. Захист BIOS паролем створює додатковий рубіж захисту від доступу до інформації. Недоліком цього способу є те, що прочитати вміст «вінчестера» можна шляхом завантаження операційної системи із зовнішнього носія, а скидання пароля BIOS теж не є важким завданням. Крім того, «вінчестер» може бути прочитаний на іншому комп'ютері.

2. Штучне руйнування або видалення файлів операційної системи теж не приводить до бажаного результату, оскільки існує велика кількість програмних засобів, що відкривають доступ до вмісту «вінчестера» у разі руйнування операційної системи.

Це завдання цілком під силу, наприклад, таким програмам, як «*ERD Commander*», «*CIA Commander*» та ін.

3. Зміна коду або знищення завантажувача IPL2, зазвичай, теж здатні перервати завантаження. Можна і взагалі повністю видалити вміст *Boot Record* завантажувального логічного диска. У цьому випадку такий диск не розпізнаватиметься навіть завантаженою ззовні операційною системою. Перенесення і підключення НЖМД до іншого комп'ютера з функціонуючою операційною системою не поліпшить ситуацію: як і раніше цей логічний диск не розпізнаватиметься, і єдиною реакцією операційної системи буде пропозиція щодо його переформатування. Проте і тут існує можливість уручну або за допомогою програм відновити вміст *Boot Record*. Текст IPL2 можна скопіювати з іншого подібного диска, а з'ясувати і прописати в Boot Record необхідні характеристики логічного диска можна за допомогою таких програмних продуктів, як *HDD Regenerator*, *mhdd*32, *Sector Inspecto*r та ін. Річ у тому, що інформація, яка міститься в *Boot Record*, є надмірною, і її можна відновити шляхом аналізу, наприклад, відповідного розділу *Partition Table* в MBR, звідки також можна з'ясувати розмір розділу.

4. З цієї ж причини руйнування *Partition Table* в MBR, хоча і призведе до тяжких наслідків, але не зможе, врешті-решт, перешкодити отримати доступ до інформації на «вінчестері». Достатньо велика кількість програм шляхом аналізу вмісту секторів «вінчестера» і пошуку за характерними сигнатурами системних розділів дають змогу не тільки визначити межі логічних дисків, але і відновити *Partition Table*. Ці завдання успішно вирішують такі програмні продукти, як *Acronis Recovery Expert Deluxe*, *Active@ Partition Recovery*, *R-Studio* та ін.

Отже, ми прийшли до висновку, що жоден з даних варіантів не дає стовідсоткової гарантії, що доступ до інформації, яку зберігали на НЖМД, не буде відновлений, а отже, захист буде подоланий. Потрібно шукати таке поєднання методів захисту, за якого переваги кожного методу будуть збережені, а недоліки — компенсуватися за рахунок застосування інших підходів.

Шифрування логічного диска

Повернемося до методів захисту, заснованих на шифруванні інформації. Є достатньо багато програмних продуктів, що виконують шифрування не тільки окремих файлів, але цілих логічних дисків. Одна з найвідоміших — програма *TrueCrypt*, яка має відкритий код і розповсюджується безкоштовно [2]. *TrueCrypt* може «на льоту» шифрувати системний розділ або цілком системний диск. За такого вигляду шифрування дані автоматично зашифровуються і розшифровуються перед їх читанням або збереженням без якої-небудь участі користувача. Файлова система при цьому шифрується в повному обсязі (шифруються імена файлів, каталогів, зміст кожного файла, вільне місце, метадані тощо). Шифрування виконується за алгоритмом *Advanced Encryption Standard* (AES), відомим, як *Rijndael*.

Шифрування системи включає аутентифікацію перед завантаженням операційної системи шляхом введення пароля. Передзавантажувальна аутентифікація здійснюється *TrueCrypt Boot Loader*, який розміщується на першому циліндрі завантажувального диска.

Таким чином, при використанні *TrueCrypt* змінюється алгоритм завантаження комп'ютера порівняно з тим, що наведено на рис. 2. Зміни стосуються передусім умов запуску завантажувача IPL2. Після старту IPL1 і пошуку координат завантажувального диска в *Partition Table* управління передається не IPL2, а *TrueCrypt Boot Loader*, за допомогою якого відбувається парольна аутентифікація користувача. Тільки при успішному виконанні аутентифікації виконується розшифровка вмісту завантажувального диска і передається управління завантажувачу IPL2.

Тобто в алгоритм завантаження комп'ютера додалася ще одна ланка — аутентифікація, яка виконується між запусками завантажувачів IPL1 і IPL2.

З метою з'ясування механізму захисту автором було проведено дослідження НЖМД, зашифрованого програмою *TrueCrypt*. З'ясувалося, що шифрування стосується тільки інформаційних розділів логічних дисків. Системні розділи як і раніше залишаються незашифрованими (рис. 3), бо інакше буде неможливо запустити *TrueCrypt Boot Loader*, оскільки його теж довелося б розшифровувати. Це дає можливість проведення атаки на *TrueCrypt* шляхом збору інформації про вміст фізичних секторів «вінчестера». Авторові вдалося без особливих зусиль виявити на зашифрованому «вінчестері» *Boot Sector* логічного диска, а також фрагменти записів з таблиці MFT, оскільки на логічному диску була розгорнена файлова система NTFS. Такі уразливості поставили під сумнів надійність використання шифрування, як єдиного методу захисту, навіть на основі передових алгоритмів криптозахисту. Підтвердження своїм сумнівам автор отримав дуже скоро, виявивши опис способу отримання пароля аутентифікації при захисті на основі *TrueCrypt* [3] та [4].

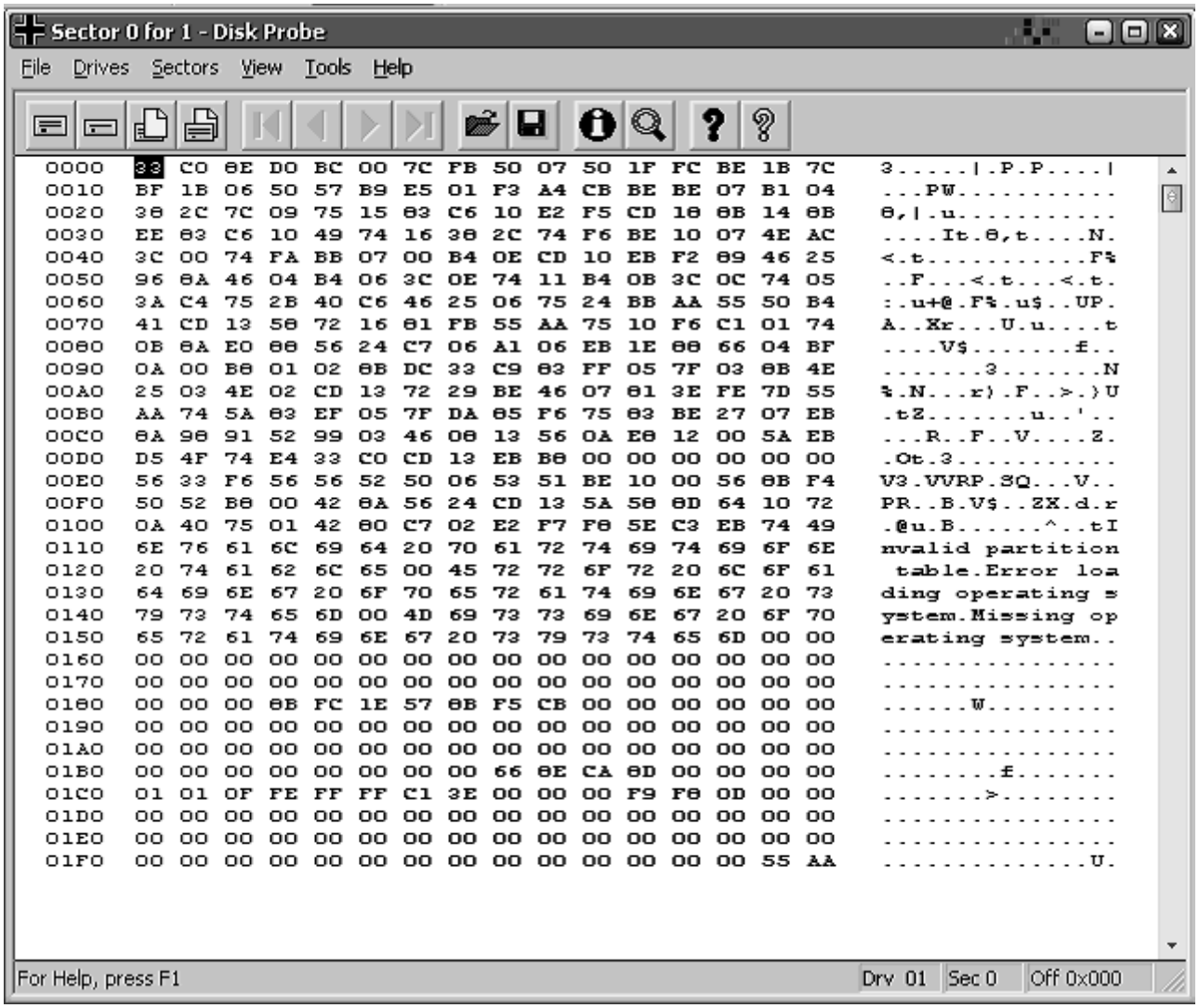

Рис. 3. *Master Boot Record* НЖМД, захищений програмою *TrueCrypt*

Виявляється, що «оперативна пам'ять комп'ютера не може вважатися абсолютно надійним сховищем секретних даних: стан ОЗУ не настільки чутливий до підтримки напруги, як вважалося раніше, і супротивник з фізичним доступом до машини здатний витягувати вміст пам'яті в початковому вигляді навіть після «холодного» перезавантаження» [4].

Незалежний фахівець з інформаційної безпеки Прінстонського університету Шеррі Давидофф пояснює, що пароль, який ввів користувач при аутентифікації в *TrueCrypt*, зберігається в кеше оперативної пам'яті в незашифрованому вигляді, що дозволяє виявити і прочитати його в процесі пошуку. Також було виявлено лічильник довжини пароля, повний шлях до нього та ім'я файла криптоконтейнера.

Сам процес розкриття пароля *TrueCrypt* був змонтований у вигляді навчального відеоролика, який опублікований на сайті *YouTube* [5], де детально і поетапно продемонстровані всі дії.

Як рекомендації розробникам подібного програмного забезпечення пропонується «проводити ретельніший аналіз власного коду, а також коду всіх загальних бібліотек і методів операційної системи, які використовуються додатком, а також обфускацию паролів в пам'яті, щоб ускладнити їх виявлення за простими текстовими підрядками» [4].

Принцип комбінованого захисту даних

Отже, методи захисту, що використовують тільки шифрування або тільки блокування доступу до «вінчестера» не приводять до успіху, оскільки кожний окремо не забезпечує надійного захисту. Проте їх комбінація дозволяє досягти поставленої мети. Пропонується побудувати захист інформації таким чином:

1. Формується завантажувальний диск на основі FLASH-пам'яті. На цьому диску, окрім файлів операційної системи розміщуються дві програми, що реалізують, відповідно, посекторні читання і запис на основі переривання BIOS INT13h.

Таке переривання забезпечує безпосереднє звернення до будь-якої кількості секторів НЖМД з подальшим читанням їх вмісту у файл або, навпаки, запис інформації з файла в певні сектори «вінчестера». Ще одна програма записує в задані сектори НЖМД байти з випадковими числами або нулями, тим самим знищуючи в них інформацію. У настройках BIOS-SETUP для завантаження з FLASH-пам'яті встановлюється найвищий пріоритет серед інших завантажувальних пристроїв.

2. Завантажувальний розділ та інші логічні диски «вінчестера» шифруються програмою *TrueCrypt.*

3. Вміст MBR і *Boot Record* всіх логічних дисків програмно копіюється на FLASH-пам'ять, після чого у всіх зазначених системних розділах інформація стирається.

У разі ввімкнення комп'ютера завантаження з «вінчестера» буде блоковано, оскільки через відсутність системних розділів він не розпізнаватиметься. Відновлення цих розділів неможливе, оскільки зашифрований вміст логічних дисків не дає змогу програмно розпізнати межі логічних дисків і файли на них. Через це неможливо проглянути кеші пам'яті на логічних дисках і знайти на них паролі *TrueCrypt*. Таким чином, доступ до вмісту «вінчестера» буде неможливий до тих пір, поки не буде реалізований такий сценарій:

— завантаження операційної системи з FLASH-пам'яті і програмне відновлення системних розділів з копій, які зберігаються на ній у файлах;

— перезавантаження операційної системи з «вінчестера» з розшифровуванням розділів *TrueCrypt* після парольної аутентифікації.

У разі завершення роботи з комп'ютером захист знову відновлюється шляхом виконання тих же дій у зворотному порядку:

- розмонтування розділів *TrueCrypt* з подальшим перезавантаженням операційної системи;
- завантаження операційної системи з FLASH-пам'яті і програмне знищення вмісту системних розділів «вінчестера».

Висновки

Поставлену мету досягнуто за рахунок застосування комбінації двох підходів до захисту інформації на НМЖД: шифрування на основі *TrueCrypt* та блокування на рівні системних завантажувачів IPL1 і IPL2.

Водночас використання тільки одного якогонебудь підходу не приведе до бажаного результату.

ЛІТЕРАТУРА

1. *Безруков Н. Н.* Компьютерная вирусология. Справочное руководство / Н. Н. Безруков. — К. : Украинская советская энциклопедия имени М. П. Бажана, 1991. — 416 с.

2. http://truecrypt.org.ua

3. http://www.sofss.ru/index.php?option=com_content&task=view&id=67&Itemid=29

4. http://forum.ru-board.com/topic.cgi?forum=5&topic=13322&start=760

5. http://www.youtube.com/watch?v=_ttbtGTlOTA

Стаття надійшла до редакції 17.12.2010..