1. Банах у к Н.В. Введение в оптимизацию конструкций. М.: Наука, 1986. 302 с. 2. Бате К., Вилсон Е. Численные методы анализа и метод конечных элементов. М.: Мир, 1982. 488 с. 3. Литвинов В.Г. Оптимизация в эллиптических граничных задачах с приложениями к механике. М.: Наука, 1987. 366 с. 4. Пелех Б.Л. Обобщенная теория оболочек. Львов, 1978. 158 с. 5. Савула Я.Г., Олейшман Н.П. Расчет и оптимизация оболочек с резными срединными поверхностями. Львов: Вища школа, 1989. 172 с. 6. Савула Я.Г., Щербатый М.В. Анализ чувствительности при оптимальном проектировании составных оболочечных конструкций // Изв. АН СССР. Механика твердого тела. 1990. Вып.1. С.137-143. 7. Хог Э., Арора Я. Прикладное оптимальное проектирование. М.: Мир, 1983. 479 с. 8. Хог Э., Чой К., Комков В. Анализ чувствительности при проектировании конструкций. М.: Мир, 1988. 428 с. 9. Bletzinger K.-U., Ramm E. Structural optimization as tool for shape design // Proc. of the First Eur. Conf. on Num. Met. in Eng., 7-11 Sept. 1992, Brussels, Belgium, 1992. P.465-467.

Стаття надійшла до редколегії 07.04.94

УДК 681.3.06

М.Ю.Щербина, П.Д.Мосорін, В.В.Черняхівський

СИНТАКСИЧНИЙ АНАЛІЗ І ЕМУЛЯЦІЯ ТРАСУВАННЯ ДЛЯ ДЕТЕКТУВАННЯ ПОЛІМОРФНИХ ВІРУСІВ

У даній праці викладені результати експериментальних досліджень вірусу OneHalf , який належить до категорій stealth і поліморфних вірусів [1, 2]. Цей вірус заражає виконавчі файли і MBR вінчестера, тобто є файлово-бутовим [1, 2]. Stealth - компонента маскує наявність бутового варіанту вірусу на вінчестері /на рівні Int 13h / і вказує стару довжину для заражених файлів /на рівні функцій 11h, 12h, 4Eh, 4Fh переривання Int 21h /. У файлах вірус закодовує своє тіло, а декодер щоразу модифікує, що і є проявом поліморфізму. Вірус не містить компонент, спрямованих на пошкодження інформації, але має компоненту, яка закодовує фрагмент вінчестера. У разі перезавантаження вірус поширює цей фрагмент і при виконанні певних умов виводить на екран фразу, яка починається так: " Dis is one half " звідки й походить неформальна назва вірусу. Закодовані сектори підставляються вірусом у нормальному вигляді /на рівні Int 13h /. Тому можливі

проблеми при завантаженні зі системної дискети і при зникненні вірусу з MBR. Остання проблема виникала у деяких користувачів, котрі виявивши вірус, відновили старий *MBR* за допомогою *ADinf* або *Norton Utilities*, унаслідок чого деяка частина інформації була втрачена. За деякими даними, вірус впливає також на роботу *Microsoft Windows*. Далі викладені загальні погляди на методи детектування поліморфних вірусів, а також побудована узагальнена модель структури декодувальної компоненти вірусу *OneHalf*.

Під терміном "вірус" розумітимемо файлові віруси або компоненти вірусів, оскільки на даний час автори не мають даних про існування бутових поліморфних вірусів, хоча для них також можна застосовувати описану нижче методику.

Як відомо [1], для детектування вірусу V у файлі F треба виділити сигнатуру $S \subset V$. Сигнатура /характеристичний код/ визначається під час аналізування коду вірусу і має бути інваріантною стосовно екземпляра даного вірусу. Сигнатура містить звичайно від 10 до 100 байтів коду, що дає змогу досить надійно і швидко детектувати вірус. При виконанні умови $S \subset F$ файл F будемо вважати зараженим вірусом V. Крім сигнатурного методу існує ще метод контрольних сум [2], але фундаментальні основи цих методів аналогічні.

Поліморфним вважатимемо вірус P, з якого ми не можемо виділити сигнатуру, тобто два екземпляри вірусу можуть цілком різнитись. Інакше кажучи, їх спільний характеристичний код може не перевищувати одного байта. Для детектування вірусу P можна розбити його на декілька вірусів $V1, V2, \ldots, Vn$ і отримати відповідні сигнатури $S1, S2, \ldots, Sn$; де n - кількість різних варіантів вірусу P, а під "вірусами" ми розуміємо всі можливі поліморфні різновиди вірусу P. Проте цей метод має недоліки, а саме:

1/ якщо n достатньо велике, що справджується для більшості поліморфних вірусів [1, 2], то питання часу роботи антивірусної програми стає критичним;

2/ деякі віруси, приміром *OneHalf*, складаються з декількох фрагментів, які не мають фіксованого місця у файлі. Хоча існують досить ефективні алгоритми пошуку підпослідовності у послідовності /файлі/, скажімо, алгоритми Кнута чи Моріса-Пратта, це не розв"язує проблему часу та ускладнює задачу "лікування".

Тут пропонується інший метод, який апробований на вірусі *OneHalf* і дав позитивні результати. Метод грунтується на тому, що інваріантним щодо модифікації.

На рис. 1 можна бачити синтаксичну діаграму декодувальної компоненти вірусу *OneHalf* /для зараженого COM-файлу, *EXE* - варіанти незначно різняться/. Початок діаграми відповідає першій машинній команді в даної програми, а кінець - останній команді декодувальної компоненти. На рис. 2 подана синтаксична діаграма для блоків коду, які генерує вірус для заповнення проміжків між функціонально значущими командами. У діаграмах використані такі позначення: *SEGCS*, *SEGDS*, *SEGES*, *SEGSS* - відповідні префікси заміни сегментів; *reg*, *reg2* - довгі регістри загального призначення; *seg* - довгий сегментний регістр, *reg1* - довгий індексний регістр; *VBeg*, *Work*, *AddV*, *VEnd* - конкретні константи, які використовуються для декодування. Робочі регістри, константи декодування і проміжковий код вірус вибирає випадковим чи-

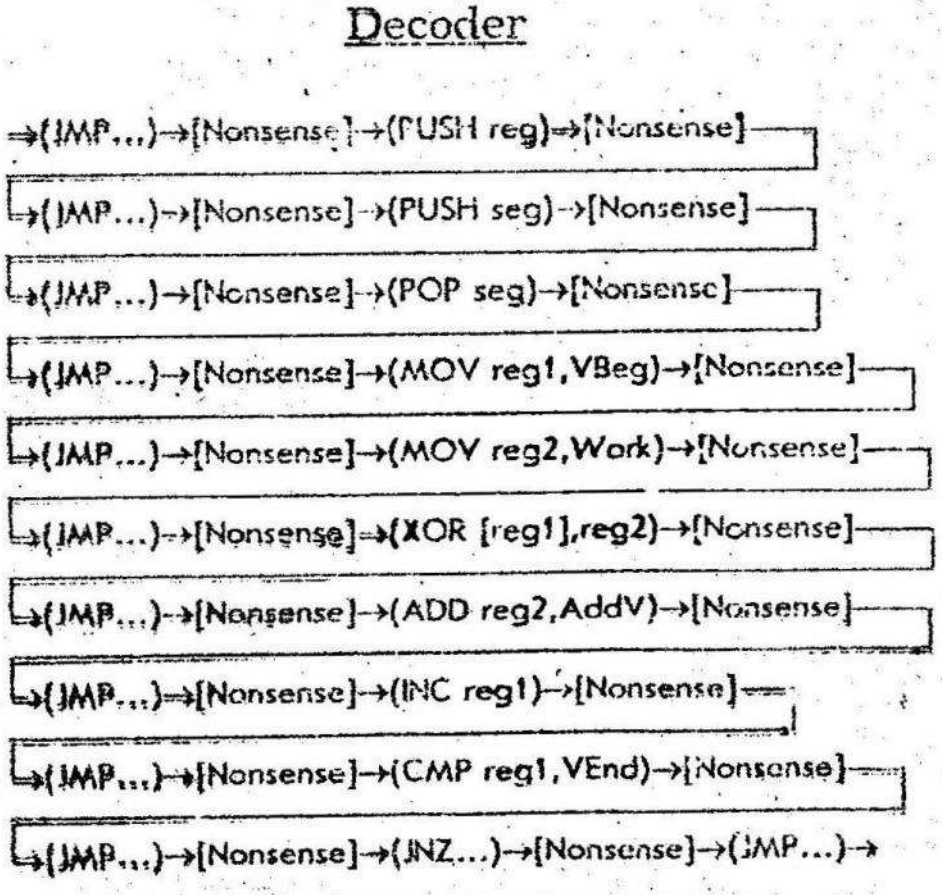

Рис. 1

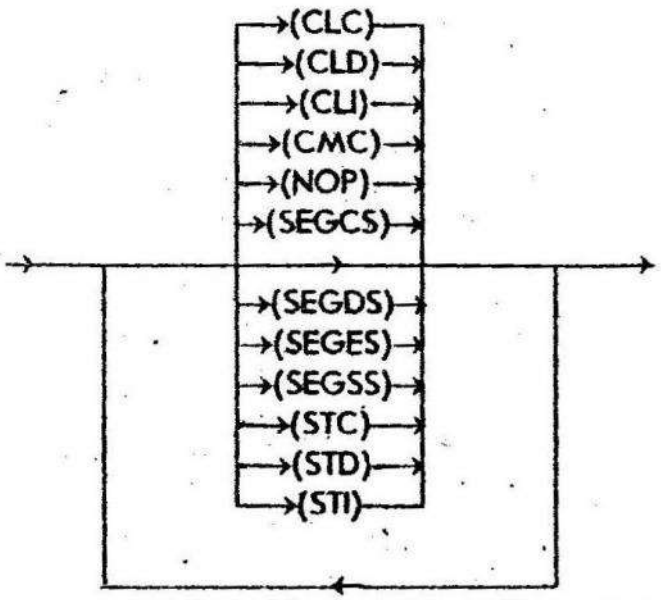

Рис. 2

```
    mov bx,[VBeg]     ;Початок закодованої частини
    mov ax,[Work]     ;Стартове значення ключа
@@1:
    xor [es:bx],ax    ;Зробити XOR з ключем
    add ax,[AddV]     ;Змінити ключ (додати AddV)
    inc bx            ;На наступний байт
    cmp bx,[VEnd]     ;Кінець закодованої частини?
    jnz @@1           ;Якщо ні, продовжити...
```

Рис. 3

ном, що і є проявом поліморфізму. Відповідно до наведених діаграм пропонується такий алгоритм роботи антивіруса.

Потрібно рухатись діаграмою /рис. 1/, емулюючи виконання команд *JMP* і "витягаючи" з деяких команд константи декодування. Для цього зручно завести змінну *IP* , функціонально аналогічну відповідному регістру процесорів 180x86 [3], і виконувати такі дії: інкрементувати *IP* для однобайтових команд, модифікувати при емуляції команд *JMP* на відповідне зміщення,

витягати за відповідною адресою константи декодування і модифікувати на довжину команди. Якщо ми "прийшли" до кінця діаграми, тобто зустріли відповідні команди у вказаній послідовності, то вважаємо, що файл заражений. Тепер відомі значення всіх констант декодування, і вірус легко декодувати /на рис. 3 поданий відповідний фрагмент коду/. Після декодування можна ще раз пересвідчитись у наявності віруса традиційними методами, оскільки він уже декодований. Можна також "вилікувати" заражений файл, оскільки вся потрібна інформація вже є доступною.

Описаний метод, на погляд авторів, можна успішно використовувати для детектування переважної більшості наявних і нових поліморфних вірусів, зокрема всіх, описаних у працях[1, 2, 7].

На основі запропонованого методу розроблена антивірусна програма, яка знаходить і знешкоджує вірус *OneHalf* у виконавчих файлах, а також, на відміну від відомих на даний час інших антивірусів, відновлює інформацію на вінчестері, що була закодована вірусом.

1. Безруков Н.Н. Компьютерная вирусология: Справочное руководство. К., 1991. 416 с. 2. Касперский Е.В. Компьютерные вирусы в *MS DOS*. М., 1992. 176 с. 3. Петрухин В.С., Степченко Ю.А., Филин А.В. Персональные ЭВМ на основе архитектуры *Intel 80386*. В 2 кн. Обнинск, 1993. Кн.1. 336 с.

Стаття надійшла до редколегії 28.10.94