Московский физико-технический институт (ГУ МФТИ)
Кафедра радиотехники

Бронированные вирусы

Эссе по курсу "Защита информации"

студент 311 группы ФРТК
Даниил Швед

МОСКВА, 2007 г.

Содержание

1 Введение

Будет рассмотрен один из случаев использования возможностей криптографии в злонамеренных целях. Подобно тому, как производители легального программного обеспечения используют эти возможности для защиты своего кода (например, защищая файлы библиотек электронной цифровой подписью). их могут использовать и создатели компьютерных вирусов. Здесь показаны общие подходы, используемые вирусописателями, и в первую очередь те из них, что полагаются на шифрование исходного кода.

2 Защита от обнаружения и анализа

Заметим предварительно, что понятие “компьютерный вирус” часто трактуется в несколько искаженном виде. Согласно представлениям большинства рядовых пользователей персонального компьютера, вирус - это произвольная вредоносная программа. В действительности под вирусом подразумевается несколько другая сущность. По определению Фрэда Коэна (Fred Cohen), профессора Калифорнийского Университета, создавшего в студенческие годы один из первых компьютерных вирусов,

Вирус — последовательность инструкций, которая при интерпретации в правильной среде меняет другие последовательности инструкций так, что создает в этой среде свою копию

По этому определению, основная цель вируса - не нанесение вреда, а распространение своих копий. Здесь уместно заметить, что часто вирусы создаются из исследовательского интереса и вообще не содержат вредоносного кода.

Большинство вирусов рано или поздно обнаруживается, и антивирусное программное обеспечение делает их распространение невозможным. По этой причине основные задачи вирусописателя - это, во первых, как можно сильнее оттянуть момент обнаружения вируса, и во-вторых, что не менее важно, затруднить анализ вируса после его обнаружения. Для достижения первой цели авторы вирусов прибегают к следующим методам:

Простейшие методы заключаются в том, чтобы сам факт заражения исполняемого файла был “не слишком заметен”. При добавлении копии вирус может проконтроллировать, чтобы даты создания и модификации исполняемого файла

сохранили прежние значения. Кроме того, часто исполняемые файлы содержат неиспользуемые области, так что вирус может добавить свой код без изменения их размера. Примерами таких вирусов (*cavity viruses*) служат "CIH" и "Чернобыль". Существуют вирусы, использующие агрессивную тактику защиты от обнаружения: они стремятся завершить работу антивирусной программы до того, как будут обнаружены. Все перечисленные техники довольно просты и неэффективны против современных систем защиты.

Избежание файлов-приманок и нежелательных объектов заражения. Эти техники следуют той идее, что менее агрессивно и интенсивно размножающийся вирус будет обнаружен позже. Во-первых, любая антивирусная программа регулярно проверяет целостность собственного кода, поэтому заражение такой программы увеличивает вероятность обнаружения вируса. Кроме того, часто системы защиты создают так называемые "файлы-приманки" (bait files) с тем, чтобы обнаружить вирус после заражения такого файла. Разумеется, вирусописатели стремятся запретить заражение таких файлов. К примеру, вирус может не выбирать в качестве очередного носителя слишком короткие запускаемые файлы или файлы с характерной закономерностью расположения "замусоревающего кода". Другая стратегия заключается в "редком заражении" (sparce infection). Скажем, принятие решения о заражении может происходить случайно.

Более серьезный метод защиты - техника "стелс" (stealth). Суть метода заключается в том, чтобы скрыть свое присутствие, перехватив некоторые запросы к операционной системе. К примеру, вирус Whale перехватывал отладочные и некоторые другие прерывания. Техника "стелс" очень мощна в том смысле, что от нее нельзя полностью обезопаситься, не загрузив систему с носителя, который с вероятностью 100% не подвергался заражению.

Теперь приведем методы затруднения анализа вируса в случае его обнаружения. В том или ином виде все эти методы связаны с изменением кода вируса в процессе его исполнения и/или создания копии. В связи с наличием такой возможности, существуют понятия вирусного набора и эволюции вируса:

Вирусный набор — набор семантически эквивалентных представлений вируса.

Эволюция вируса — его действия по превращению из одного представления в другое в одном вирусном наборе.

Шифрование кода вируса. При использовании этого подхода вирус состоит из двух частей: зашифрованный код вируса и небольшой расшифровывающий модуль. Обычно применяются симметричные схемы шифрования, а ключ меняется каждый раз при создании вирусом своих копий. Тем не менее, ключ вместе с процедурой расшифрования содержится в открытом виде в неизменной части вируса, поэтому нет никаких препятствий к его расшифрованию. Кроме того, расшифровывающий модуль один и тот же у всех экземпляров вируса, поэтому такой вирус легко обнаруживается при помощи сигнатур.

Полиморфизм - это расширение идеи шифрования кода. Отличие заключается в том, что расшифровывающий модуль изменяется от модуля к модулю, не меняя при этом своей функциональности. Полиморфный код - основа создания брони-

рованных вирусов, и используемые в нем приемы будут подробнее рассмотрены ниже.

Метаморфизм . Вирус полностью изменяет собственный код при создании копии. Метаморфные вирусы обычно чрезвычайно сложны и, следовательно, достаточно велики, причем большую часть вируса составляет собственно код, отвечающий за метаморфизм (metamorphic engine). Вот некоторые из методов, используемых при копировании:

- Перемешивание кода. Изменение порядка слдедования процедур и инструкций.
- Изменение последовательности исполнения. В отличие от перемешивания, в этом случае в код вставляются команды перехода (например, jmp и call для архитектуры x86).
- Представление одних и тех же команд разными способами. Наиболее популярные процессоры для персональных компьютеров содержат сильно избыточный набор команд, поэтому некоторые инструкции дают один и тот же результат (скажем, `"mov ax, 0h"` и `"xor ax, ax"`).
- Замусоривающие инструкции. Можно просто вставить бесполезные инструкции среди полезных.
- Использование различных регистров для выполнения одних и тех же действий в разных версиях в пределах одного вирусного набора.

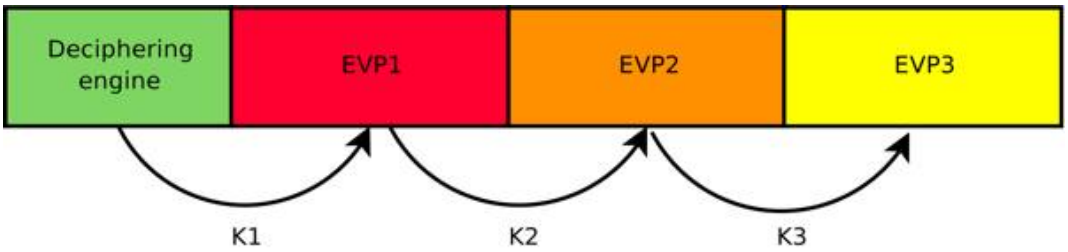
3 Бронированные вирусы

Бронированным называют вирус, использующий технологии предотвращения или задержки анализа своего кода. Таким образом, из рассмотренных техник к собственно бронированию относятся полиморфизм и метаморфизм.

Моментом осознания потенциальной опасности компьютерных вирусов, серьезности их угрозы и необходимости борьбы с ними считают 1990 год. В этом году появился и распространился уже упомянутый вирус Whale, в котором использовались технологии стелс, полиморфизма, и некоторые другие методы бронирования (запутывание кода, зависимость изменения кода от архитектуры, анти-отладчик). Из всего перечисленного для нас наибольший интерес представляют поли- и метаморфизм.

Уже упомянутая техника полиморфизма обладает одним очевидным недостатком: хотя ключи шифрования вместе с процедурой расшифрования и меняются от версии к версии, расшифрованный код вируса всегда один и тот же. Поэтому код вируса легко можно обнаружить в памяти процесса уже после расшифрования. Эта проблема решается сочетанием техник полиморфизма и метаморфизма. Именно, metamorphic engine создает всякий раз новую версию кода вируса, а polymorphic engine создает новую версию модуля, расшифровывающего этот код.

Рис. 1: Структура вируса Bradley



Мутация кода хорошо действует против обнаружения вируса, но не так эффективна против его анализа. Действительно, в рассмотренном подходе имеется следующий недостаток: ключ шифрования содержится в расшифровывающем модуле в открытом виде, поэтому для аналитика не представляет сложности расшифровать основной код. Поэтому следующая идея - вообще удалить значение ключа из тела вируса. Часто вирусописатели применяют очень простые (но зато быстрые и компактные) процедуры шифрования, например XOR. В этом случае часто представляется возможным получить значение ключа в результате криптоанализа. Впрочем, на это тоже требуется время, и его может оказаться достаточно, чтобы вирус распространился достаточно для целей его создателя. Иначе обстоит дело при использовании надежных алгоритмов шифрования.

3.1 Bradley

Вирус Bradley считается не поддающимся анализу. Структура вируса (рис. 3.1) состоит из 4 частей:

- Дешифрующий модуль. Генерирует ключ и расшифровывает остальные три части.
- `EVP1`[1] — содержит анти-антивирусные механизмы.
- `EVP2` — отвечает за реализацию поли- и метаморфизма, а также за распространение вируса.
- `EVP3` — любой другой код

Последние три части шифруются на трех различных ключах: k_1, k_2 и k_3. "Надежность" Bradley основывается на надежности защиты этих ключей. Разумеется, вирус, несущий в себе ключ шифрования, не может не поддаваться анализу. В Bradley расшифровывающий модуль (deciphering engine) генерирует ключ в зависимости от того, в какой среде распространяется вирус. Под "средой" могут пониматься, к примеру такие данные:

- Системное время
- Конфигурация системы

[1] EVP - Environmental Viral Payload, "начинка" вируса в определенной среде.

- Погода, значения курсов на рынке валюты
- Хеш-значение определенной страницы Web
- Содержимое какого-либо файла
- и т.д.

Обозначим условно n характеристики среды, на которые полагается вирус.

Кроме этого, для создания правильных ключей Bradley нужна информация, которая находится под контролем вирусописателя. Назовем ее a.

Блок `EVP1` шифруется с ключом $k_1 = H(a + n)$. Далее `EVP2` шифруется с $k_2 = H(k_1 + c_1)$, где c_1 - последние 512 байт незашифрованного блока `EVP1`. Аналогично $k_3 = H(k_2+c_2)$. В расшифровывающей процедуре хранится значение $m = H(H(k_1+ c_1) + e_1)$, где e_1 - первые 512 байт зашифрованного `EVP1`. Расшифровывающий модуль получает ключи следующим образом:

1. Собирает информацию о среде n, а также информацию, доступную создателю вируса, a.
2. Вычисляет значение $H(H(a+n)+e_1)$, и сравнивает его с известным значением m. Если равенства нет - полностью удаляет весь вирус из системы.
3. В случае равенства вычисляет первый ключ $k_1 = H(a+n)$ и расшифровывает `EVP1`.
4. Последовательно получает ключи и расшифровывает `EVP2` и `EVP3`.

При самовоспроизведении вирус меняет не только код всех четырех частей, но и значения ключей, а также значение m. Для этого достаточно обновить значение n. Исходя из нового значения n' будет вычислен новуй ключ k'_1, соответственно изменятся и все остальные ключи.

Если предположить, что функция H криптографически надежна, то Bradley нельзя проанализировать, даже если он будет обнаружен. Значение a нельзя извлечь из кода вируса, так как последний содержит только хеш от него. Это свойство дает вирусописателю следующие преимущества:

- Невозможно определить, на какие системы нацелен вирус.
- Можно настроить вирус на определенную жертву, используя, к примеру, e-mail адрес, если это человек, или другую специфичную для жертвы информацию.

4 Заключение

Рассмотренный вирус Bradley иллюстрирует основные идеи бронирования вирусов. Существуют и дргие идеи, которые или продолжают показанные, или предлагают новые, в определенном смысле более дерзкие подходы к использованию криптографии в неблаговидных целях. Примером такой идеи может служить отсутствие не только ключа, но и процедуры расшифрования в теле вируса. В самом

деле, ничто не мешает вирусу использовать в этих целях средства пораженной системы, скажем, CryptoAPI в системе Windows. При разработке любой системы, использующей криптографию, необходимо иметь подобные возможности в виду. Важно не "переигрывать", и наряду с сохранностью данных беспокоиться также и об открытости системы, так как излишняя скрытность может быть только на руку злоумышленнику.

Список литературы

[1] "Криптография: палка о двух концах, часть 2", securitylab.ru.
http://www.securitylab.ru/analytics/288278.php

[2] "Computer virus", Wikipedia.
http://en.wikipedia.org/wiki/Computer_virus

[3] "Компьютерный вирус", Wikipedia.
http://ru.wikipedia.org/wiki/Компьютерный_вирус

[4] "Полиморфизм компьютерных вирусов", Wikipedia.
http://ru.wikipedia.org/wiki/Полиморфизм_компьютерных_вирусов