Системный подход к реверс-инжинирингу программных продуктов обрабатывающих векторные данные.

Докладчики:

Кавешников Максим Борисович

доцент МИИГАиК и Московского Университета МВД.

Махина Екатерина Константиновна

зам.декана МИИГАиК.

Современные компьютерные мошенники пока нападают в основном на банковские системы, но скоро они освоят и системы работающие с геопространственными данными.

Реверс – инженеры это программисты высшего класса, которые способны выделить инородные вставки в программный код.
Реверс-инжиниринг – обратное проектирование – по готовому изделию нужно восстановить технологию и техническую документацию.
Законодательно реверсинг допустим, если подвергаемое ему изделие получено законным путём. **Цель легального реверсинга** – контроль соответствия предоставленного изделия заявленным свойствам.

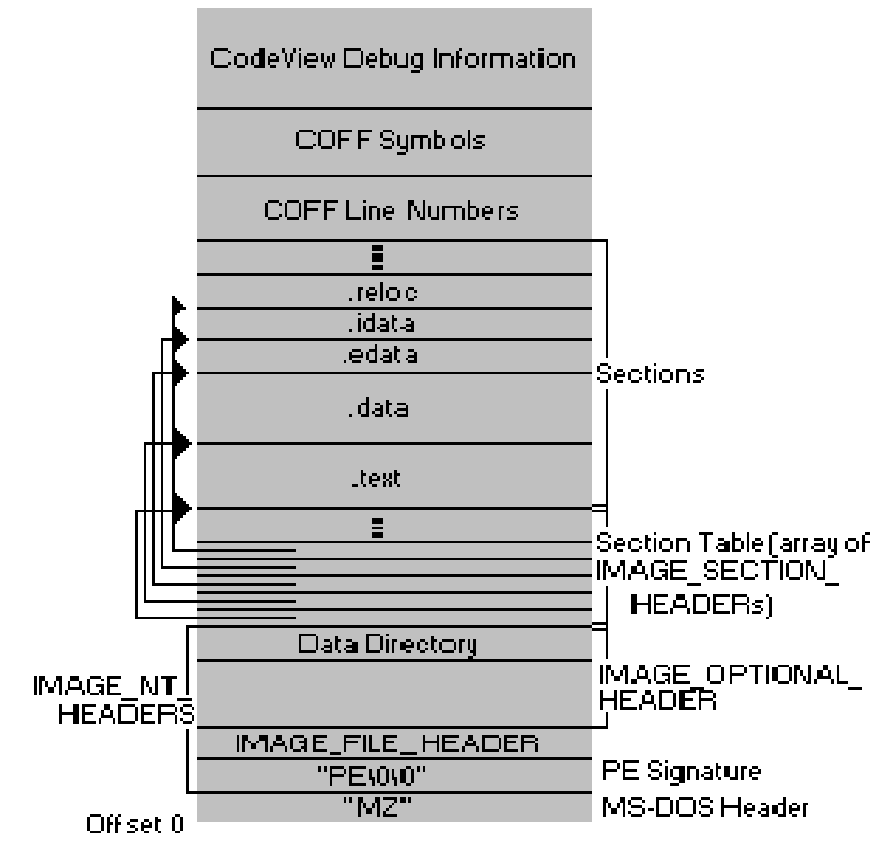
Традиционный подход к реверс-инжинирингу

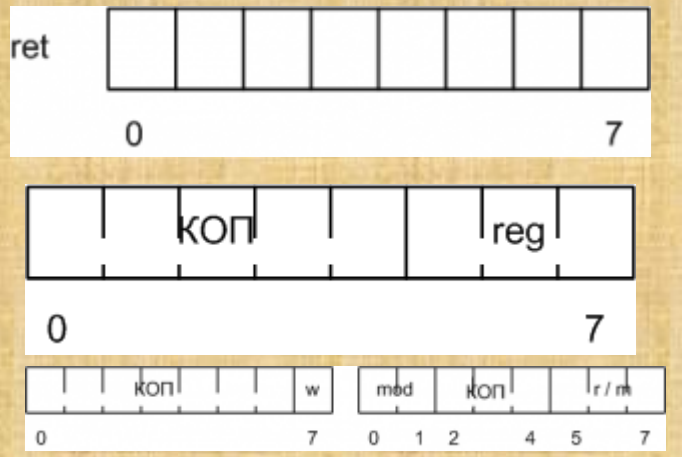
- Дисассемблирование
- Внешний отладчик
- Декомпиляция
- Анализ программы на ассемблере или на языке высокого уровня
- В случае открытого кода анализируется открытый исходный код
- Анализ устанавливает характер действия кода.
- Обычно результат анализа –результат понимания кода человеком.
- Вредный или несанкционированный код выделяется по действию, которое отличается от назначения программы.

Разнообразие способов несанкционированного внедрения кода

- 1. Затирание - размещение внедряемого кода **поверх текста оригинальной программы.**
- 2. Интеграция – размещение внедряемого кода **в свободном месте исполняемого файла** оригинальной программы.
- 3. Дописывание - запись в начало, середину или конец файла с сохранением оригинального содержимого.
- 4. Размещение **вне файла оригинальной программы** с размещением внедряемого кода в динамической библиотеке, или в NTFS потоке. Загрузка при этом выполняется малым загрузчиком («головой кода»), который может быть внедрён способом 1, 2, 3.
- **Полиморфизм** – возможность иметь многовариантный код, с выбором варианта в момент выполнения.
  Самоидентифицирующийся код – код который сам себя преобразует в момент выполнения программы. Создаёт трудности для дезасемблирования, поскольку дезасемблер работает со статическим снимком (дампом) памяти или диска.

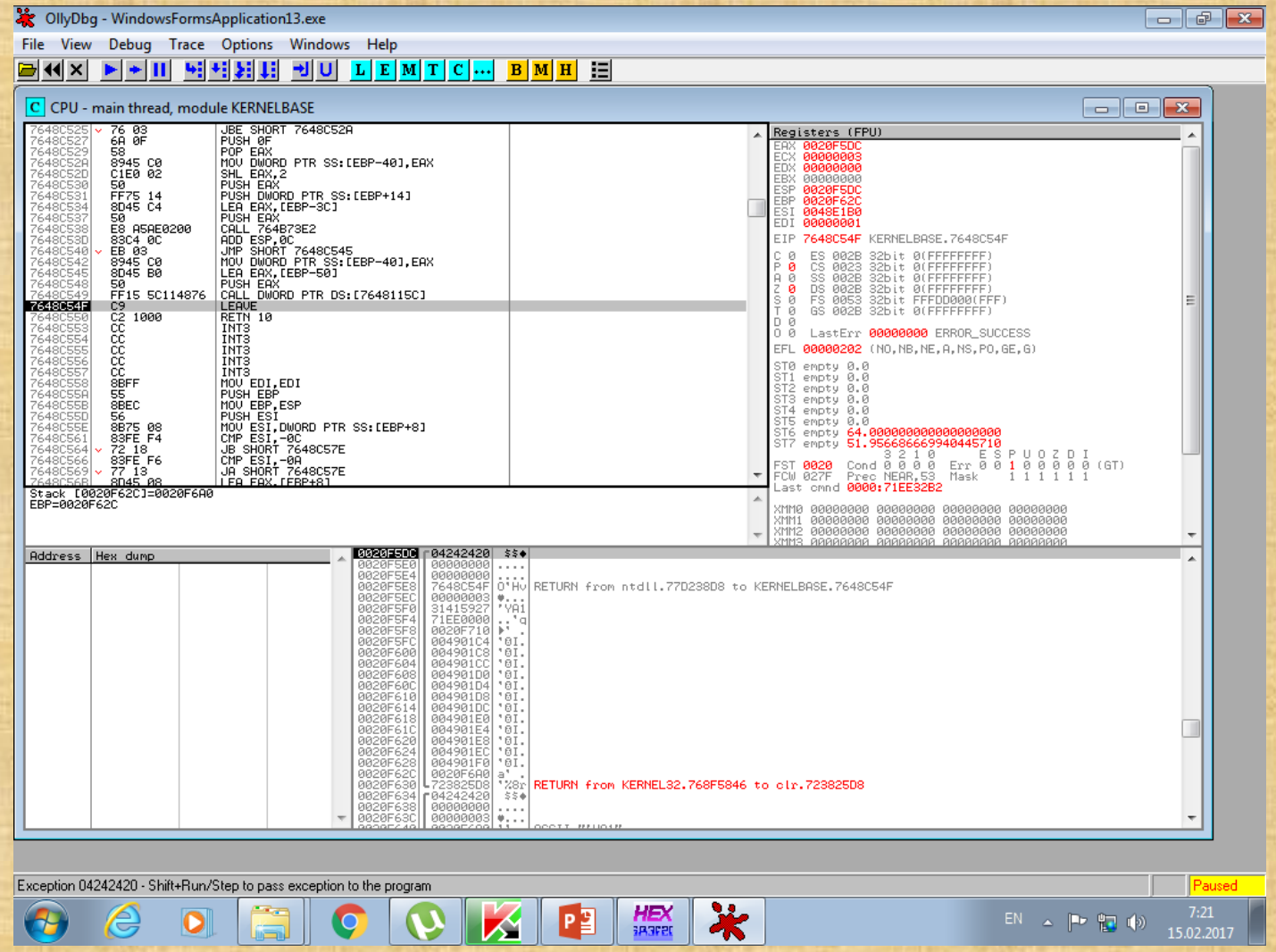
Формат исполняемого файла и формат машинных команд.





mod = 11
r/m - код регистра
w = 0 - 8-разрядный регистр
w = 1 - 16-разрядный регистр

Внешние отладчики и дизассемблеры



Антивирусные программы

- Поиск характерных участков кода – патернов или сигнатур.
- Создание баз данных сигнатур
- Создание баз кодов вирусов
- Ручное удаление вирусов самое эффективное

  Поиск по сигнатурам – вирус как правило не должен повторно себя внедрять в один и тот же файл. Поэтому, он должен в своём коде иметь метку, идентифицирующую его наличие. Эту метку так же может выделить и антивирус и даже, если код вируса сильно модифицируется. Эту метку называют **сигнатурой**.
- Вирусы написанные так, что могут внедряться несколько раз встречаются редко, так как их разработка сложнее.

Особенности пространственных данных и обрабатывающих программ

- Большие массивы однородных данных обрабатываемых однотипными алгоритмами
- Топологические связи между данными, возможность их использования при восстановлении
- Характерные приёмы экономии памяти и вычислительных операций.
- Стандартизированные хранилища данных.
- Характерный интерфейс.
- Наличие определённых числовых значений связанных с геопривязкой

Все это может быть дополнительной информацией для разделения продуктивного кода и вредной закладки.

Специализированные утилиты!

Способы защиты программ от исследования

- ***Запутывание или обфускация кода*** – процесс приведения кода к трудному для анализа виду, при условии сохранения его функциональности. Обычно это замусоривание неисполняемыми фрагментами.
- ***Вычисление контрольных сумм или проверка целостности кода*** – периодический подсчёт контрольных сумм от определённых участков оперативной памяти, занимаемой кодом программы.
- ***Шифрование отдельных элементов кода и данных***-для защиты закрытых для пользователя элементов программы выполняется зашифровывание как данных, так и текстов программ. Это сильно затрудняет исследование кода. Обычно шифруют данные, которые организованы определённым образом, например, данные хранящиеся в стеках.

Системный подход

- Математическая модель компьютера может быть задана в виде **линейной динамической системы с дискретным временем :**
- $\overrightarrow{X_{к+1}} = \Phi(t_{k+1}|t_k)\overrightarrow{X_k} + B_k(\overrightarrow{V_k} + \overrightarrow{\xi_k})$, (1)
- где $\overrightarrow{X_к}$ - **вектор состояния** на момент t_k, $\Phi(t_{k+1}|t_k)$ - **переходная матрица** вектора состояния с момента t_k на момент t_{k+1} , $\overrightarrow{V_k}$ **– вектор документированных воздействий,** $\overrightarrow{\xi_k}$ - **вектор неучтённых (недокументированных) воздействий** на состояние компьютера. B_k - матрица, характеризующая изменение состояния компьютера вследствие управляющих воздействий. Наблюдатель за состоянием компьютера не может полностью иметь информацию о векторе $\overrightarrow{X_к}$. Наблюдению подлежат лишь некоторые производные от этого вектора состояния параметры. **Уравнения наблюдения будет:**
- $\overrightarrow{Y_k} = C_k\overrightarrow{X_к}$ (2)
- C_k – **матрица наблюдения**. Она описывает связь между параметрами, которые пользователь может вывести из компьютера и состоянием компьютера. Вектор $\overrightarrow{V_k}$ и матрица B_k могут быть заданы таким образом, что $\overrightarrow{V_k}$ может рассматриваться как компьютерная программа, а $\overrightarrow{\xi_k}$ – посторонний, неописанный в документации код.

Диаграммы переходов состояний (STD).

Рис. 6.1: Символы STD

Начальное состояние - узел STD, являющийся стартовой точкой для начального системного перехода. STD имеет только одно начальное состояние, соответствующее состоянию системы после ее инсталляции, но перед началом реальной обработки.

Состояние может рассматриваться как условие устойчивости для системы. Находясь в определенном состоянии, мы имеем достаточно информации о прошлой истории системы, чтобы определить очередное состояние в зависимости от текущих входных событий. Имя состояния должно отражать реальную ситуацию, в которой находилась - система, например, НАГРЕВАНИЕ, ОХЛАЖДЕНИЕ и т.п.

Карта допустимых адресов и передач управления.

Исполняемые файлы имеют три вида адресации:

Физические адреса или смещения (offset) – отсчитываются от начала файла, виртуальные адреса (virtual address VA) – отсчитывается от начального адреса пространства процесса, относительные виртуальные адреса (relative virtual address RVA) - отсчитываемые от базового адреса загрузки (начальный адрес модуля внутри процесса). Все адреса относительные. Практически можно построить диаграммы передач управления от одного адресного пространства другому. У программ выполняющих векторные вычисления эти передачи и адреса наиболее предсказуемы.

Реализация в виде специального отладчика, прилагаемого к продукту

Состав отладчика:

Виртуальная машина – эмулятор

Карта передачи управлений

Модуль сравнения передач управлений и адресного пространства эмулятора и карты.

Реализация в виде свободно-скачиваемых утилит

Или в виде удостоверяющего центра.

Авторы, преподаватели,

студенты и выпускники

МИИГАиК и МОСУ МВД

благодарят компании GPScom и

Йена Инструментс за

многолетнюю помощь, участие и

поддержку!