

Учитель Начальной

Курс Реверс инжиниринг информационных систем

Учитель Начальной

Курс Реверс инжиниринг

информационных систем

<https://litres.ru/74599012>

SelfPub; 2026

Аннотация

Учебное пособие посвящено реверс-инжинирингу информационных систем. В двадцати главах последовательно рассматриваются основы обратного проектирования программного обеспечения, правовые и этические аспекты, архитектура процессоров, язык ассемблера, форматы исполняемых файлов PE и ELF, инструменты статического и динамического анализа. Отдельные разделы посвящены антиотладке и обфускации, анализу сетевых протоколов, веб-приложений, мобильного ПО, баз данных, вредоносных программ и прошивок встраиваемых систем. Приводятся практические примеры, команды инструментов, типовые паттерны кода и лабораторные задания. Заключительные главы охватывают автоматизацию анализа, современные тенденции и профессиональное развитие. Пособие предназначено для студентов и специалистов, изучающих информационную безопасность и сопровождение сложных программных систем.

Содержание

Предисловие	4
Глава 1. Введение в реверс-инжиниринг информационных систем	6
1.1. Определение и сущность	6
1.2. Цели и задачи	7
1.3. Классификация	8
1.4. Типичный workflow реверс-инженера	9
1.5. Пример первого взгляда на файл	10
Глава 2. Правовые, этические и организационные аспекты	11
2.1. Правовая база	11
2.2. Этические принципы	12
2.3. Организация безопасной лаборатории	13
2.4. Чек-лист перед анализом	14
Глава 3. Архитектура компьютера и основы низкоуровневого программирования	15
3.1. Процессоры x86/x64	15
3.2. Соглашения о вызовах	16
Конец ознакомительного фрагмента.	17

Учитель Начальной Курс Реверс инжиниринг информационных систем

Предисловие

Настоящее расширенное учебное пособие представляет собой полноценный курс по реверс-инжинирингу информационных систем. Материал рассчитан на специалистов и студентов, имеющих базовые знания программирования (желательно C/C++), операционных систем и сетей.

Реверс-инжиниринг - процесс анализа готовой системы с целью понимания её устройства, алгоритмов и структуры данных без доступа к исходным материалам. В контексте информационных систем это включает анализ программного обеспечения, протоколов, архитектур, баз данных и встраиваемых решений.

В этом издании каждая глава дополнена практически примерами, типичными командами инструментов, распознаваемыми паттернами кода, советами и предупреждениями.

Важное предупреждение: все практические работы выполняйте только в законных рамках - на собственных про-

граммах, учебных crackme, открытых malware-семплах в изолированной среде. Несанкционированный анализ чужого программного обеспечения может нарушать законодательство.

Рекомендуемая продолжительность изучения - 4-6 месяцев при регулярных занятиях. Необходимы: виртуальная машина, базовые знания C/C++ и готовность разбирать низкоуровневые детали.

Глава 1. Введение в реверс-инжиниринг информационных систем

1.1. Определение и сущность

Реверс-инжиниринг (reverse engineering) - исследование объекта с целью выявления принципов его работы путём анализа конечного продукта. В отличие от прямого проектирования, реверс идёт от готовой системы к пониманию её внутреннего устройства.

Применительно к информационным системам это анализ исполняемых файлов, библиотек, протоколов, схем баз данных, архитектурных решений и конфигураций без исходных кодов и документации.

1.2. Цели и задачи

Основные цели:

- Анализ безопасности и поиск уязвимостей
- Исследование вредоносного программного обеспечения
- Документирование и понимание legacy-систем
- Модернизация и миграция устаревших информационных систем
- Обеспечение совместимости и интероперабельности
- Восстановление утраченной документации
- Образовательные и исследовательские цели

1.3. Классификация

По уровню: битовый и бинарный, уровень исходного кода (декомпиляция), архитектурный, уровень данных и бизнес-логики.

По методу: статический (без выполнения), динамический (во время выполнения), гибридный.

По объекту: software reverse engineering, protocol reverse engineering, hardware и firmware reverse engineering, data reverse engineering, architecture recovery.

1.4. Типичный workflow реверс-инженера

1. Сбор информации об объекте (тип файла, строки, импорты, хеши).
2. Настройка изолированной среды.
3. Статический анализ (дизассемблер, декомпилятор).
4. Динамический анализ (отладчик, мониторы).
5. Документирование находок (алгоритмы, структуры, ИОС).
6. При необходимости - написание скриптов и повтор итераций.

1.5. Пример первого взгляда на файл

Команды для быстрой разведки (Linux):

```
file sample.exe
```

```
strings -n 8 sample.exe | head -50
```

```
sha256sum sample.exe
```

На Windows удобно сразу открыть файл в Detect It Easy или PE-bear, чтобы увидеть компилятор, упаковщик и секции.

Глава 2. Правовые, этические и организационные аспекты

2.1. Правовая база

В Российской Федерации реверс регулируется нормами авторского права (ГК РФ, часть IV), законодательством об информации и уголовным законодательством. Законный реверс возможен при анализе собственного ПО, open-source, в образовательных и научных целях, а также при анализе malware специалистами по информационной безопасности с соблюдением установленных процедур.

Обход технических средств защиты авторских прав без разрешения правообладателя, как правило, незаконен. Всегда проверяйте лицензию и цель анализа.

2.2. Этические принципы

- Работать только с разрешёнными объектами
- Не распространять результаты анализа проприетарного

ПО без разрешения

- Изолировать вредоносный код и не запускать его на рабочих системах

- Уважать права владельцев интеллектуальной собственности

- Документировать действия в профессиональной деятельности

2.3. Организация безопасной лаборатории

Рекомендуемый минимум:

- Хост-машина с VirtualBox или VMware
- Несколько виртуальных машин: Windows 10/11, Linux (Ubuntu/Debian)
- Снимки (snapshots) перед каждым рискованным действием
- Host-only или полностью изолированная сеть
- Общая папка только для передачи семплов

Для malware-анализа дополнительно можно использовать Cuckoo Sandbox или облачные сервисы с разрешённым использованием.

2.4. Чек-лист перед анализом

- Есть ли разрешение, это мой файл, open-source или учебный семпл?
- Среда изолирована? Есть snapshot?
- Сеть отключена или контролируется?
- Антивирус на хосте не мешает работе инструментов?

Глава 3. Архитектура компьютера и основы низкоуровневого программирования

3.1. Процессоры x86/x64

Ключевые регистры x64: RAX (аккумулятор, возвращаемое значение), RBX, RCX, RDX, RSI, RDI, RBP (base pointer), RSP (stack pointer), R8-R15. Младшие части: EAX, AX, AL и аналогичные.

Флаги (RFLAGS): ZF (zero), SF (sign), CF (carry), OF (overflow) - используются условными переходами (JE/JZ, JNE, JA, JB, JG, JL и другими).

Стек растёт вниз. Вызов функции: аргументы (по соглашению), return address, сохранённые регистры, локальные переменные.

3.2. Соглашения о вызовах

Microsoft x64 calling convention (Windows): первые 4 аргумента в RCX, RDX, R8, R9; остальные в стеке; возвращаемое значение в RAX.

System V AMD64 (Linux): RDI, RSI, RDX, RCX, R8, R9; возврат в RAX.

Знание calling convention критично для понимания, где искать аргументы функции при отладке и статическом анализе.

Конец ознакомительного фрагмента.

Текст предоставлен ООО «Литрес».

Прочитайте эту книгу целиком, [купив полную легальную версию](#) на Литрес.

Безопасно оплатить книгу можно банковской картой Visa, MasterCard, Maestro, со счета мобильного телефона, с платежного терминала, в салоне МТС или Связной, через PayPal, WebMoney, Яндекс.Деньги, QIWI Кошелек, бонусными картами или другим удобным Вам способом.