Нанесение водяных знаков на программное обеспечение

Архипов Иван Сергеевич

Санкт-Петербургский Государсвенный Университет

группа 21.M04-мм
Научный руководитель: д.ф-м.н., профессор А.Н.Терехов
Консультант: старший преподаватель Уральского федерального университета, А.Е.Сибиряков

Обзор водяных знаков

- Статические водяные знаки
- Динамические водяные знаки

Характеристики водяных знаков

- Надёжность
- Требуемый объём ресурсов
- Невидимость
- Защита частей кода, а не всего проекта целиком
- Устойчивость
- Ортогональность

Новая архитектура

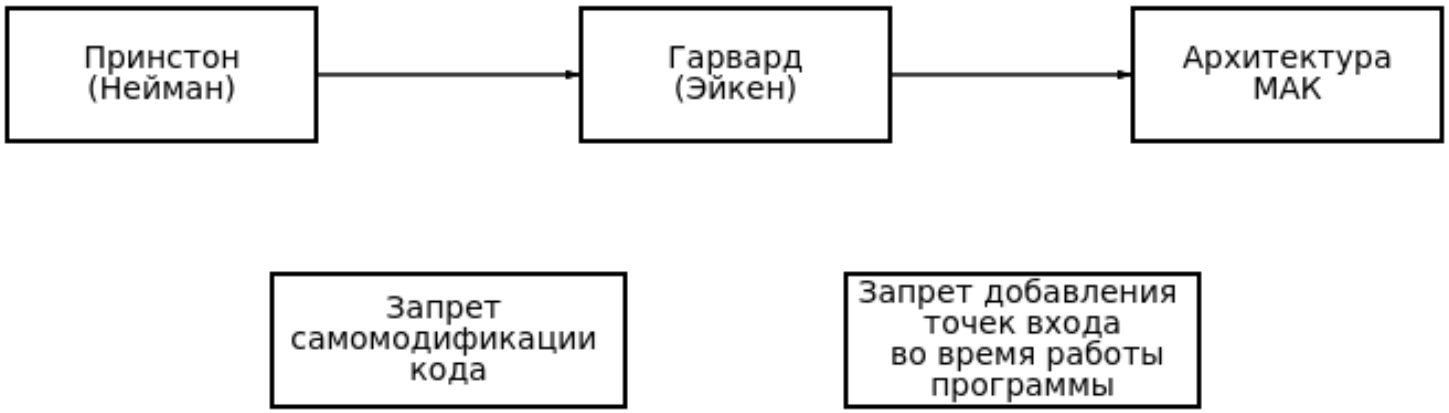

Рис.: Архитектурные концепции

Новые возможности

- В Гарвардской архитектуре нельзя определить точки входа программы. Из-за этого нельзя “раздвинуть” линейные участки кода и перемешать их, так как мы не можем заранее сказать, придёт ли управление в данную точку
- В новой архитектуре этого недостатка нет, что открывает широкие возможности, например, для запутывания кода. В том числе в данной архитектуре открываются новые возможности для нанесения водяных знаков на программное обеспечение

Водяные знаки в новой архитектуре

- Можно “раздвинуть” линейные участки кода, а в них записать водяной знак. Способ нанесения водяного знака зависит от его цели. Например, если нужен видимый водяной знак, то можно просто его туда записать, если нужен невидимый, то нужен способ его спрятать
- Хочется иметь способ наносить целый комплекс водяных знаков с разными характеристиками

Каркасный водяной знак

- Необходим способ понимать, что в данной области записан водяной знак. Этот способ и есть каркасный водяной знак
- На данный момент предлагается использовать редкие битовые последовательности. Но такие последовательности легко найти, потому хочется иметь способ их спрятать

Вероятностные характеристики ПО

- Оказалось, что программный код имеет вполне определённые вероятностные характеристики. Например, эмпирически было выяснено, что нулей в программном коде примерно 60 %, а единиц 40 %
- Чтобы спрятать водяной знак, нужно “подогнать” эмпирические статистики такие, как у обычного кода. Для этого нужно эти статистики найти
- Также для сокрытия водяного знака можно использовать такие “редкие” последовательности, которые не всегда встречаются в исполняемом файле. Для этого нужно провести анализ частоты нахождения различных битовых последовательностей в коде

Предшествующая работа

Данная тема рассматривалась в курсовой работе Смирнова Дениса Павловича в 2018 году.

- Был сделан хороший обзор водяных знаков
- Вычислены некоторые статистики, однако недостаточное количество
- Отсутствие анализа
- Отсутствие готовой спецификации по нанесению водяных знаков

Цель и задачи

Целью работы является создание спецификации нанесения водяных знаков на программное обеспечение. Для достижения обозначенной цели были посталены следующие задачи:

- Освоение стека технологий
- Обзор водяных знаков
- Подготовка данных для нахождения статистик и редких битовых последовательностей
- Нахождение статистик и редких битовых последовательностей
- Анализ метода нанесения водяного знака
- Написание спецификации

Извлечение секции кода для анализа

- В elf (Executable and Linkable Format) файле имеется множество секций с самой различной информацией: данные для работы программы, данные об исполняемом файле, программный код
- Для нахождения статистик необходимо уметь извлекать секцию кода .text из elf файла

Алгоритм извлечения секции кода [1]

1. Получить ссылку на section header table (смещение 0x28 от начала файла). В section header table хранится различная информация о секциях. Запись с информацией о каждой секции занимает 10 байт
2. Получить индекс секции .shstrtab в section header table (смещение 0x3E от начала файла). Секция .shstrtab хранит название секций, ссылки на которые хранятся в section header table
3. Идти по section header table по секциям. По смещению (смещение 0x00 от начала записи) на .shstrtab определить, информация о какой секции хранится в данной записи таблицы
4. Если дошли до записи о секции .text, то получить смещение секции от начала файла (смещение 0x18 от начала записи) и её размер (смещение 0x20 от начала записи). Можно извлекать данные

[1] Весь код для подготовки и анализа данных расположен в репозитории: `https://github.com/IvanArkhipov1999/Binary-code-statistics-research`

Выбор данных для анализа[2]

Для сбора информации о программном коде необходимо выбрать данные для анализа. Были рассмотрены следующие варианты:

- Все исполняемые файлы на компьютере. Сторонний читатель не может повторить эксперименты, трудности с определением принадлежности файла к elf формату, только одна архитектура
- Несколько крупных проектов. Даёт мало информации о распределении статистик ввиду малой выборки
- Тестовая база Clang. Не пригодны для сборки и получения elf-файла
- Тестовая база GCC. Содержит около 4000 файлов, пригодных для сборки

[2]Весь анализируемый датасет лежит в отдельном репозитории `https://github.com/IvanArkhipov1999/Binaries-dataset`

Анализ данных

- Выбрать какую-либо характеристику кода и получить выборку
- Попробовать "угадать" распределение и вычислить его параметры по методу максимального правдоподобия. Проверить гипотезу принадлежности выборки данному распределению по критерию согласия Колмогорова
- При невозможности "угадать" распределение вычислить как можно больше статистик выбранной характеристики: среднее, среднеквадратичное отклонение, мода и другие

Поиск редких последовательностей

- В качестве характеристик кода были выбраны доли битовых последовательностей различной длины. Это позволит найти и статистические характеристики кода, и редкие битовые последовательности
- В качестве примера для анализа были изучены исполняемые файлы следующих архитектур: x86_64, arm64 и mips64el
- Рассмотрены все возможные битовые последовательности длины от 1 до 10 включительно. Найдены все битовые последовательности длины до 10 включительно, которые попадаются не во всех бинарных файлов из датасета

Результаты для архитектуры x86_64

Последовательность	Среднее	Стандартное отклонение
0	0.5934	0.0302
1	0.4066	0.0302
00	0.3945	0.0323
01	0.1989	0.0164
10	0.1988	0.0164
11	0.2077	0.0363

Таблица: Доли последовательностей для архитектуры x86_64

Результаты для архитектуры x86_64

Последовательность	Среднее	Стандартное отклонение
000	0.2667	0.0379
001	0.1279	0.012
010	0.1316	0.0182
011	0.0674	0.0037
100	0.1278	0.0121
101	0.071	0.0052
110	0.0673	0.0037
111	0.1404	0.0339

Таблица: Доли последовательностей для архитектуры x86_64

Результаты для архитектуры x86_64

Последовательность	Среднее	Доля файлов
10110100	0.003	0.333%
11100110	0.00023	15.614%
110101100	0.0008	3.878%
111001101	5.275e-05	45.917%
1111001101	2.205e-05	65.1%
1111111011	0.0011	0.026%

Таблица: Редкие последовательности для архитектуры x86_64

Всего было найдено 508 редких последовательностей длиной 8, 9 и 10 бит. В таблице указана доля файлов, не содержащих последовательность.

Результаты для архитектуры arm64

Последовательность	Среднее	Стандартное отклонение
0	0.6284	0.033
1	0.3716	0.033
00	0.4646	0.0446
01	0.1637	0.0126
10	0.1637	0.0126
11	0.2078	0.0226

Таблица: Доли последовательностей для архитектуры arm64

Результаты для архитектуры arm64

Последовательность	Среднее	Стандартное отклонение
000	0.3684	0.0505
001	0.0962	0.0074
010	0.092	0.0074
011	0.0718	0.0068
100	0.0962	0.0074
101	0.0676	0.0068
110	0.0718	0.0068
111	0.1361	0.017

Таблица: Доли последовательностей для архитектуры arm64

Результаты для архитектуры arm64

Последовательность	Среднее	Доля файлов
00110011	0.0003	9.759%
11001101	0.0028	0.026%
010000110	0.0009	4.314%
110110010	8.041e-05	36.954%
1011011101	1.311e-05	68.079%
1111001101	0.0024	0.103%

Таблица: Редкие последовательности для архитектуры arm64

Всего было найдено 517 редких последовательностей длиной 8, 9 и 10 бит. В таблице указана доля файлов, не содержащих последовательность.

Результаты для архитектуры mips64el

Последовательность	Среднее	Стандартное отклонение
0	0.7	0.014
1	0.3	0.014
00	0.5587	0.0143
01	0.141	0.0031
10	0.141	0.0031
11	0.1952	0.0148

Таблица: Доли последовательностей для архитектуры mips64el

Результаты для архитектуры mips64el

Последовательность	Среднее	Стандартное отклонение
000	0.4605	0.0128
001	0.0981	0.0032
010	0.0778	0.0048
011	0.0632	0.0036
100	0.0981	0.0032
101	0.0429	0.0027
110	0.0632	0.0036
111	0.0959	0.0115

Таблица: Доли последовательностей для архитектуры mips64el

Результаты для архитектуры mips64el

Последовательность	Среднее	Доля файлов
10011010	9.528e-05	19.516%
10101101	2.532e-05	49.665%
000011101	0.0003	0.772%
011111101	0.0002	5.149%
1010101101	8.503e-07	93.46%
1100110010	0.0001	10.453%

Таблица: Редкие последовательности для архитектуры mips64el

Всего было найдено 406 редких последовательностей длиной 8, 9 и 10 бит. В таблице указана доля файлов, не содержащих последовательность.

Дальнейшие планы

- Анализ метода нанесения водяного знака
- Написание спецификации