УДК 004

АНАЛИЗ ВРЕДОНОСНОГО КОДА С ПОМОЩЬЮ ИНТЕРАКТИВНЫХ ИНСТРУМЕНТОВ ДИЗАССЕМБЛЕРА И СОСТАВЛЕНИЕ СРАВНИТЕЛЬНОГО АНАЛИЗА

Рашитов Нұралбек Сексенбайұлы
nurashura5808@gmail.com
магистрант, факультет информационных технологий
ЕНУ им. Л.Н.Гумилева, Нур-Султан,Казахстан
Научный руководитель – Г. Бекманова

Абстракт

В мире, в котором мы живем, существует бесчисленное множество программных приложений, и каждый день создаются новые. Кроме того, большинство из них имеют скрытый исходный код, что приводит к большей работе при попытке понять специфику, алгоритмы и т.д. Инструмент обратного проектирования используется для деконструкции двоичного файла, чтобы выявить такие знания, как дизайн программ, архитектура или даже найти уязвимости. Моя цель в этом проекте – сравнить эти инструменты обратного инжиниринга. В частности, я стремлюсь проанализировать GHIDRA и IDA Pro. Каждый из этих инструментов имеет свою собственную реализацию того, как работать с обратным инжинирингом, и задача здесь состоит в том, чтобы выявить эти систематические различия и подготовить подробный отчет о том, какие из этих инструментов лучше в различных аспектах. На основе анализа я подготовил метрики для оценки каждого из них и дадим рекомендации, основанные на наших результатах.

1. Описание проекта

В этом проекте я рассмотрел приложения обратного инжиниринга, а именно IDA Pro и GHIDRA. Каждое из этих приложений является хорошо известным программным обеспечением. Однако сравнение затруднено для не авторов из-за систематических различий. Поэтому в этом проекте моя цель обеспечить справедливое сравнение между этими двумя инструментами. Для этого мы рассмотрели список сравнительных метрик [1]. Эти показатели следующие: GUI-UI/UX, графики потоков управления, поддерживаемый язык, поддерживаемая архитектура и отказоустойчивость. Это поможет нам лучше оценить сильные и слабые стороны каждого программного обеспечения. Для тестирования этого программного обеспечения мы рассмотрели различные программы, написанные на нескольких языках. У меня есть короткая средняя и большая программа на каждом языке, это будет обсуждаться далее в специальном разделе для каждого инструмента.

2. Предыстория

Инструменты реверс-инжиниринга полезны во многих сферах. Они полезны для инженеров-программистов и сопровождающих, чтобы понять структуру программного обеспечения путем анализа исходного кода и представить их на более высоком уровне

абстракции с помощью графов потоков управления и графов вызовов. Без помощи таких инструментов чрезвычайно трудно построить их для двоичных файлов большого сложного программного обеспечения.

3. Оценка возможностей инструментов

3.1 IDA Pro – это размещенный в Windows, Linux или MacOS X многопроцессорный дизассемблер и отладчик, что означает, что он поддерживает несколько целей отладки и может обрабатывать удаленные приложения через «удаленный сервер отладки». Однако в этом проекте мы сосредоточимся только на дизассемблере IDA. IDA лицензирована и продается по очень высокой лицензионной цене, но она поставляет так много функционалистов, что исследователи безопасности и бинарного анализа рассматривают возможность оплаты ее Pro edition.

3.1.1 Поддерживаемые архитектуры. IDA Pro поддерживает более 50 различных архитектур, что очень впечатляет между этими тремя инструментами, которые мы проанализировали. Ниже приведен краткий список поддерживаемых архитектур, однако полный список можно найти здесь: Архитектура x64 (Intel x64 и AMD64),архитектура ARM64 (она же AArch64), Dalvik (Android bytecode, DEX), DEC Alpha.

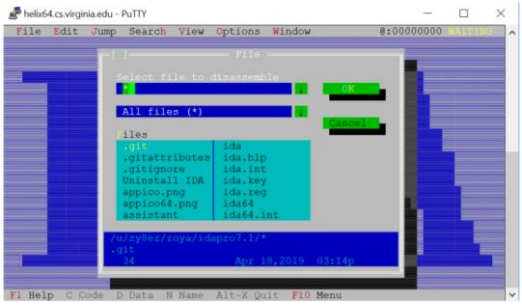

Рис. 1. Пользовательский интерфейс IDA Pro

3.1.2 Поддерживаемые типы файлов. IDA Pro способна разбирать практически любой популярный формат файлов. Краткий список этих форматов можно найти ниже. Однако полный список можно найти здесь: MS DOS, EXE-файл, MS DOS COM-файл, драйвер MS DOS, двоичный файл, ZIP-архив, архив JAR

3.1.3 Поддерживаемые языки. Чтобы определить, способен ли каждый из наших инструментов разбирать и анализировать различные файлы, я собрал несколько различных программ, написанных на разных языках программирования, а затем запустили их против целевых инструментов. Для этой цели мы выбрали следующие языки программирования: C, C++, Perl, Python, Fortran, Ruby, Shell и exe-файлы. IDA Pro успешно читала, разбирала и анализировала программы, написанные на всех этих языках.

3.1.4 GUI-I/UX. IDA Pro – это один из инструментов, который, как говорят, имеет сравнительно очень удобный пользовательский интерфейс. Однако у меня был доступ к IDA Pro через машину Helix64. Чтобы запустить IDA Pro на этой машине, нам сначала нужно было подключиться к ней через SSH, а затем запустить IDA из терминала. Это привело к тому, что графический интерфейс выглядел совсем не так, как я ожидал увидеть. Этот пользовательский интерфейс очень похож на пользовательский интерфейс приложений MS-DOS (рис.1), поэтому он не очень удобен для пользователя. Кроме того, исследуя IDA Pro, я обнаружил, что изначально он поставляется с таким количеством различных панелей инструментов и функций, которые отсутствовали в графическом интерфейсе, с которым я работал. По этой причине, основываясь на наших тестах и анализе, графический интерфейс IDA Pro менее желателен по сравнению с GHIDRA.

3.1.5 Блок-график. IDA Pro рекламирует функцию flow graphing, которая была причиной, по которой я решил исследовать и сравнить эту функцию среди инструментов, которые я сравниваю. IDA Pro поддерживает построение графиков через порт VCG, откуда она может создавать стандартные графики GDL, которые затем передаются в Wingraph32 для

рисования. Wingraph32, частичный порт графической библиотеки VCG, которая доступна начиная с IDA Используя этот графический интерфейс, мы смогли сгенерировать файл GDL, но в нем отсутствовали опции для рисования графика.

3.1.6 Отказоустойчивость. Чтобы измерить, насколько отказоустойчив каждый из инструментов, мы изменили несколько байтов в двоичном файле с помощью команды «dd» и проверили, может ли инструмент обнаружить и проанализировать файл. IDA Pro успешно прочитала поврежденный файл без каких-либо сообщений об ошибках и смогла извлечь из него некоторую информацию. Тем не менее, я отмечаю, что извлеченная информация из поврежденного файла была не полностью правильной, но она была точной по большей части.

3.2 GHIDRA – это бесплатный инструмент обратного инжиниринга с открытым исходным кодом, разработанный агентством национальной безопасности (АНБ). Он написан на C++ и Java. Двоичные файлы были выпущены на конференции RSA в марте 2019 года, источники были опубликованы месяцем позже на GitHub. GHIDRA оснащена для работы с несколькими сценариями, а также способна выполнять многие вещи, такие как анализ скомпилированных кодов на различных платформах, включая Windows, Mac OS и Linux, Разборка, сборка, декомпиляция, построение графиков и сценариев, работа в интерактивном и автоматизированном режимах. Пользователи могут разрабатывать подключаемые компоненты GHIDRA и/или скрипты с использованием открытых API.

3.2.1 Поддерживаемые архитектуры. GHIDRA поддерживает множество архитектур, и многие новые добавляются в новые версии каждый день. Относительно новое программное обеспечение и, следовательно, еще не поддерживает так много архитектур. Список поддерживаемых архитектур приведен ниже: x86/Linux, ARM and AARCH64, PowerPC 32/64, PowerPC VLE.

3.2.2 GUI - UI/UX. GHIDRA имеет очень красивый и очень интуитивно понятный дизайн [рис.2]. Окно разделено на несколько компонентов, а именно

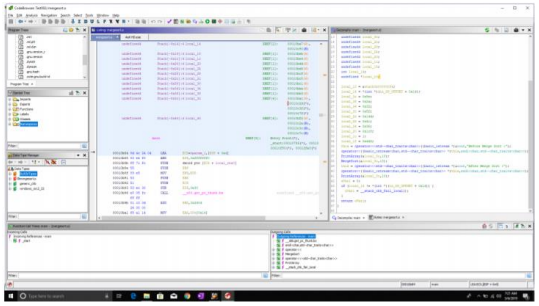

Рис. 2. Пользовательский интерфейс GHIDRA

- Ассемблерный код. В этом разделе мы видим машинный код из скомпилированного файла. Это показывает нам код, который помогает нам найти различные компоненты и общую логику выполнения.
- Окно декомпилятора. В этом разделе мы видим псевдокодовое представление функций на языке Си. Это помогает нам получить информацию об используемых переменных, магических числах, типах данных, а также об используемой логике.

3.2.4 Блок-график. GHIDRA позволяет вам работать над блок-графиком и браузером кода одновременно. Удивительно, что окно графика функций и окно браузера кода синхронизированы друг с другом. Используя два экрана, мы можем пройти через поток кода в режиме графика на одном экране и сохранить другой экран для ассемблерного кода, который полезен, когда мне нужны детали в ассемблерном коде или для добавления комментариев и переименования меток и т.д. Мы также можем проследить связи между функциями и то, как передается управление; кроме того, существование циклов и рекурсий облегчается.

3.2.5 Поддерживаемые типы файлов. GHIDRA поддерживает широкий спектр форматов файлов, как уже упоминалось для IDA Pro.

3.2.6 Отказоустойчивость. GHIDRA работает очень хорошо в случае сбоев, мы повредили скомпилированный файл и попытались перепроектировать его с помощью GHIDRA. Результаты были удивительными, так как файл, уменьшенный с 7096 КБ до 116 КБ, дает отличные результаты, поскольку файл не только открыт, но и проанализирован. Анализ поврежденного файла и исходного файла аналогичен, и поэтому мы можем сделать вывод, что отказоустойчивость очень хороша.

4. Обсуждения

4.1 Сильные стороны

Все инструменты являются очень мощными и универсальными инструментами для реверс-инжиниринга и бинарного анализа. Однако у каждого из них есть свои сильные стороны, которые делают их уникальными и желательными для конкретной задачи, которую вы хотите выполнить.

4.1.1 IDA Pro. У IDA Pro есть отладчик, а у GHIDRA -нет. Если ваше приложение нуждается в отладчике, то вы должны предпочесть IDA Pro. Множество доступных плагинов (например, IDAng). IDA существует уже долгое время, и поэтому у нас есть огромное количество доступных плагинов. Поддержка огромного количества платформ; IDA Pro поддерживает более 50 различных архитектур, что очень впечатляет по сравнению с другими инструментами, которые мы проанализировали. Большая поддержка со стороны разработчиков и их форумов, потому что он используется в течение более длительного времени, чем GHIDRA.

4.1.2 GHIDRA. GHIDRA - это бесплатное программное обеспечение с открытым исходным кодом, которое делает его легким для всех. Мы можем добавить личные модули, чтобы лучше помочь нам.

GHIDRA позволяет нескольким реверс-инженерам совместно использовать проект – это делает реверс-инжиниринг совместным подходом и, следовательно, помогает в непрерывном мониторинге, а также обновлении двоичных файлов.

Undo option. Эта опция была добавлена в GHIDRA и позволяет отменить определенные части файла, чтобы разрешить динамическое редактирование. Контроль версий помогает лучше поддерживать двоичные файлы.

GHIDRA позволяет добавлять несколько файлов в один и тот же проект.

GHIDRA заметно быстра для файлов размером более 1 ГБ.

4.2 Ограничения

4.2.1 IDA Pro. IDA Pro не имеет кнопки отмены – кнопка отмены помогает вносить динамические изменения в файл, не беспокоясь о повреждении двоичного файла. Еще одним серьезным ограничением IDA Pro является его цена. Этот инструмент очень дорогой по сравнению с его новым конкурентом, GHIDRA, который поставляется бесплатно.

4.2.2 GHIDRA. У GHIDRA нет такого отладчика, как IDA Pro. Кроме того, GHIDRA - довольно новый инструмент, поэтому, в отличие от IDA Pro, для него еще не разработано много плагинов. Кроме того, документация для него не полностью разработана, что приводит к задержке в изучении GHIDRA.

5. Заключение

Исходя из моих оценок и использования каждого из инструментов, я могу сказать, что ни один из них не является лучшим среди них. У каждого из них есть свои сильные и слабые стороны. Все они преуспевают в том, чтобы быть хорошим инструментом обратной инженерии, но каждый из инструментов может предложить нечто большее, чем другие. Кроме того, у нас есть еще много платформ и языков, используемых во многих реальных приложениях, которые у нас не было возможности проанализировать, поскольку они не соответствовали нашим временным ограничениям этого проекта. Однако мои усилия в этом проекте заключались в том, чтобы провести объективное сравнение между этими

инструментами с использованием короткого списка сравнительных показателей, чего я успешно достиг.

Список использованных источников

1. B Bellay and H Gall. 1997. Сравнение четырех инструментов реверс-инжиниринга. Материалы 4-й Рабочей конференции по реверс-инжинирингу, 2-11. https://doi.org/10.1109/WCRE.1997.624571

2. Ронни Шевалье, Стефано Кристалли, Кристоф Хаузер, Ян Шошитаишвили, Руою Ван, Кристофер Крюгель, Джованни Винья, Данило Бруски и Андреа Ланци. BootKeeper: Проверка свойств целостности программного обеспечения в Образах загрузочного микропрограммного обеспечения. В КОДАСПИ. ACM, 315-325.

3. Николас Нетеркот и Джулиан Сьюард. 2007. Valgrind: фреймворк для тяжеловесной динамики

бинарное приборостроение.. В PLDI, Жанна Ферранте и Кэтрин С. Маккинли (Ред.). ACM, 89-100. http://dblp.uni-trier.de/db/conf/pldi/pldi2007.html#NethercoteS07

4. Ян Шошитаишвили, Руоюй Ван, Кристофер Саллс, Ник Стивенс, Марио Полино, Одри Датчер, Джон Гросен, Сидзи Фэн, Кристоф Хаузер, Кристофер Крюгель и Джованни Винья. Сок: (Состояние) Искусство войны: Наступательные приемы в бинарном анализе. На симпозиуме IEEE по безопасности и конфиденциальности.

5. Джейкоб Спрингер и У-чан Фэн. Обучение с помощью angr: Учебная программа символического исполнения и CTF. В ASE @ USENIX Security Symposium. Ассоциация USENIX.