**Проектирование ОС и компонентов**

Обфускация, обработка кода заданными методами и применение и сравнение заданных и выбранных обфускаторов.

**Теоретическая часть**

**Что такое Обфускация?**

Обфуска́ция (obfuscate — делать неочевидным, запутанным, сбивать с толку) или запутывание кода — приведение исходного текста или исполняемого кода программы к виду, сохраняющему её функциональность, но затрудняющему анализ, понимание алгоритмов работы и модификацию при декомпиляции.

«Запутывание» кода может осуществляться на уровне алгоритма, исходного текста и/или ассемблерного текста. Для создания запутанного ассемблерного текста могут использоваться специализированные компиляторы, использующие неочевидные или недокументированные возможности среды исполнения программы. Существуют также специальные программы, производящие обфускацию, называемые обфускаторами.

Цели обфускации:

Затруднение декомпиляции/отладки и изучения программ с целью обнаружения функциональности.

Затруднение декомпиляции проприетарных программ с целью предотвращения обратной разработки или обхода DRM и систем проверки лицензий.

Оптимизация программы с целью уменьшения размера работающего кода и ускорения работы.

## Обфускатор ProGuard

Одним из самых известных обфускаторов является [ProGuard](http://proguard.sourceforge.net/). Изначально это была библиотека для оптимизации кода, поэтому он не настолько мощный как некоторые платные обфускаторы на подобии Stringer. Но в сочетании с тем же Stringer'ом благодаря ProGuard можно добиться просто хороших результатов.

ProGuard позволяет делать следующее:

* минификация и сжатие скомпилированного кода;
* непосредственно обфускация;
* обнаружение и удаление ненужного кода;
* оптимизация кода.

Воспользовавшись Proguard можно уменьшить размер скомпилированного приложения в несколько раз, что до сих пор актуально, например, для мобильных приложений.

Для написания своего обфускатора надо уметь модифицировать байт-код скомпилированных классов. Существуют библиотеки, позволяющие это делать, например, ASM. Если дойдут руки, в будущем вместе с вами напишем простой обфускатор java кода своими руками. Ну а на сегодня пока что все.

## Обфускатор yGuard

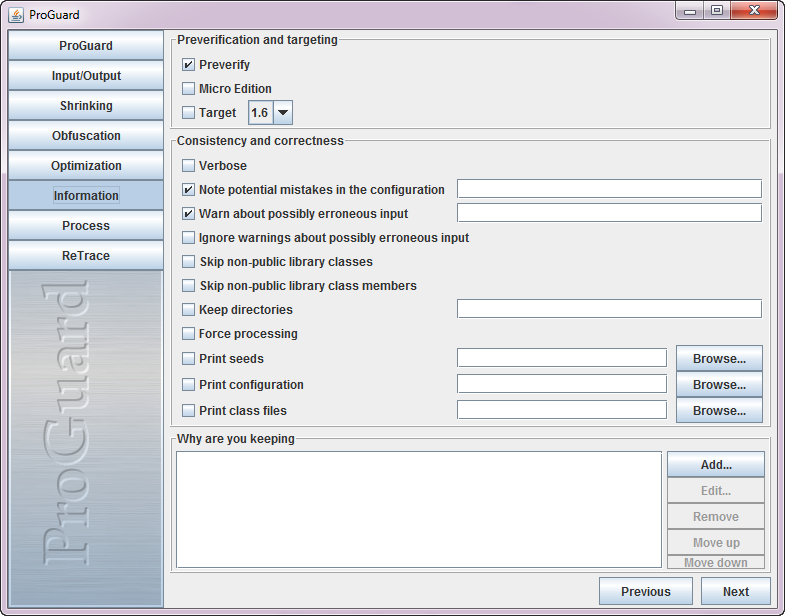
Используя задачу yGuard Ant, обфускацию и код можно легко интегрировать в процесс развертывания. Задача yguard содержит два вложенных элемента, которые выполняют обфускацию имени и сокращение кода отдельно: Усадочный элемент удаляет все элементы элементов, которые недоступны из точек входа, указанных во вложенном элементе keep. Элемент переименования выполняет обфускацию имен, переименование всех пакетов, классов, методов и полей в соответствии с выбранной схемой сопоставления имен. Элементы можно исключить из процесса переименования, аннотируя их определенным классом аннотаций в исходном коде или используя вложенный элемент keep. Оба элемента работают с одним и тем же набором файлов ввода и вывода, указанных в содержащем элементе yguard.

**ProGuard:**

Код защищен и обфусцирован с помощью Proguard. Для удобства была использована GUI-версия Proguard. Были подменены имена классов, призведена оптимизация и усложнена возможность реинжинеринга кода.

Обфускация:

Создано в IDE Eclipse запускаемый .jar файл. Использовано программа Proguard GUI. Запускается > Во вкладке Input/Output > Нажимается Add Input Добавил jar файл с программой > Добавляется Output file, его местоположение и название > Выбрается в Obfuscation такие опции, как: обфускация, агрессивная перегрузка, изменение имен классов. После нажатие Process.



Рисунок

Получается готовый обфусцированный файл.

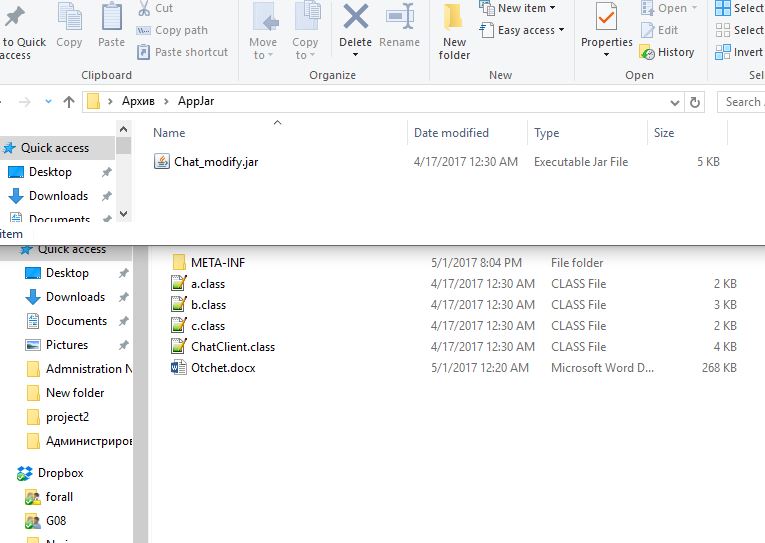
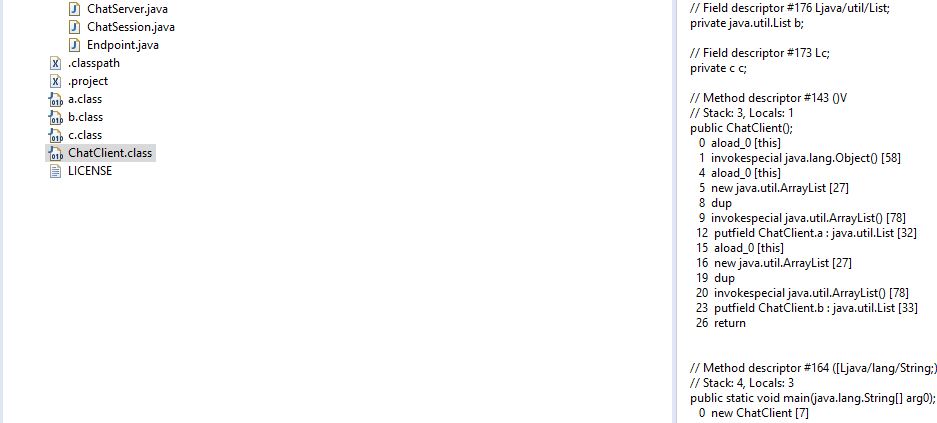


Рисунок 2



Рисунок

Декомпилировал исполняемый .jar файл онлайн-ресурсом.



Рисунок

**JFuscator:**

Для Обфускации также можно использовать *JFuscator.*

Для открытие программы используется строка написано внизу

|  |
| --- |
| Java –jar jfuscator.jar |

Затем открывается, выбирается исходный код программы и делается build.

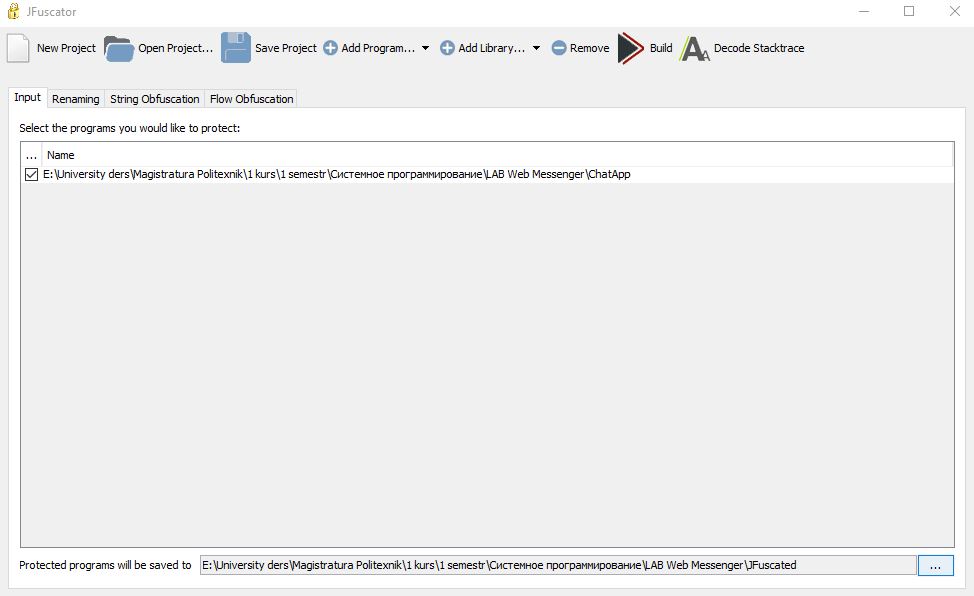


Рисунок 5

Результат обфускации и зашиты с помощью JFuscator. Создается папка bin (class).

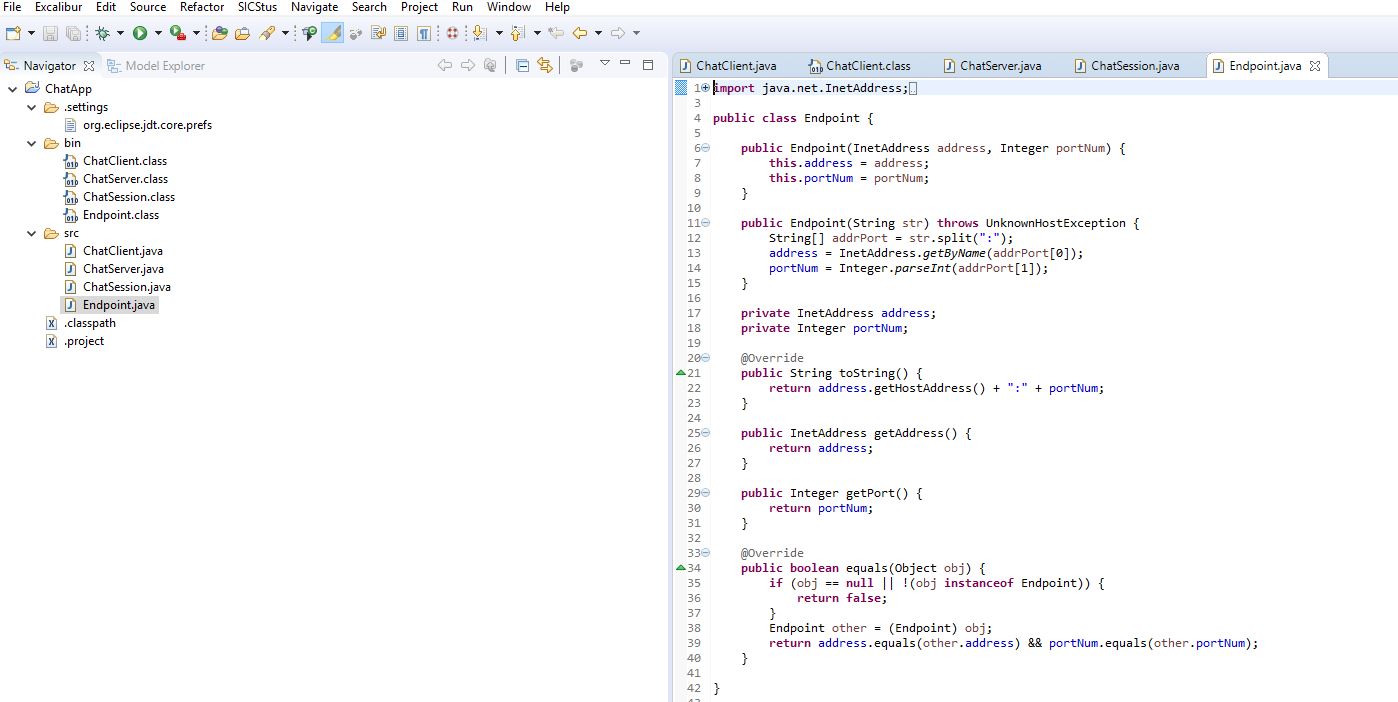


Рисунок 6

**Список используемых источников:**

1. Декомпиляция - <http://www.javadecompilers.com>
2. Программа для Обфускации - <https://www.guardsquare.com/en/proguard>
3. <https://sourceforge.net/directory/os:windows/?q=java+obfuscator>