ные сайты не будут работать. Таким образом, наиболее верным решением будет использовать в качестве headless браузера самый популярный браузер в мире - Google Chrome.

На данный момент существует библиотека Puppeteer, которая имеет очень удобный API с большой документацией по управлению headless Chrome browser. Для работы с Puppeteer требуется установленные утилиты node js и npm. В этом случае с помощью библиотеки можно имитировать такие действия пользователя, как движения мышкой, нажатия кнопок, ввод данных в формы ввода, ожидания и задержки. Puppeteer имеет наиболее полную документацию и широкую поддержку, что делает ее одним из лучших плагинов по работе с headless браузерами.

Примеры использования библиотеки Puppeteer для обработки информации из различных источников обсуждаются в докладе.

А.В. Черенко (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **А.И. Кучеров,** ст. преподаватель

АРХИТЕКТУРА КОМПЛЕКСА ПО ПРЕДОТВРАЩЕНИЮ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА К УЗЛУ ЛВС

В текущий момент информация имеет большую ценность, поэтому первоочередной задачей является обеспечение безопасности данных. Стойкость подсистемы аутентификации определяется гарантией того, что злоумышленник не сможет пройти процедуру подтверждения личности. Для повышения стойкости, помимо классических систем аутентификации, можно использовать технологию мониторинга рабочей нагрузки, порождаемой пользователем. А также процедуры проверки пользователя на соответствие с заранее составленным портретом.

Программный комплекс по получению информации о используемых приложениях и времени их работы в операционной системе будет снимать различные показатели с вычислительной системы. Исходя из этих данных можно будет судить о характере рабочей нагрузки создаваемой пользователем в вычислительной системе, а также классифицировать рабочую нагрузку.

В настоящее время существуют решения, позволяющие отслеживать рабочую нагрузку, однако данные решения не способны прини-

мать меры по борьбе со злоумышленником, например, отстранение пользователя от системы или ограничение его доступа в системе.

Проект состоит из нескольких модулей. Клиентский модуль сбора данных это приложение, никак не влияющее на работу пользователя, и собирающее статистику активности последнего. Собранные данные передаются серверной части для анализа и обработки, на этом этапе формируется портрет пользователя. Анализатор входных данных, при расхождении с заранее составленным портретом или при внезапном обрыве мониторинга сообщит об этом пользователю. Далее в зависимости от заданных параметров проводится повторная аутентификация или ограничение доступа пользователя.

Для функций администрирования отведен отдельный механизм который позволяет проводить настройку параметров системы дополнительной аутентификации, создавать новые и корректировать уже имеющиеся портреты пользователей, регулировать уровни доступа пользователей и тд.

А.В. Черенко (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **А.И. Кучеров,** ст. преподаватель

РАЗРАБОТКА КОМПЛЕКСА ПО ПРЕДОТВРАЩЕНИЮ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА К УЗЛУ ЛВС

Среди основных этапов разработки можно выделить проектирование, разработка логики приложения, разработка _Unit-тестов, тестирование системы, внедрение приложения в узел локальной вычислительной системы.

Программа написана на языке объектно-ориентированного программирования _Java. Для тестирования использовалась библиотека TestNG, также приразработке испольковался контейнер сервлетов Apache Maven.

При реализации приложения использовались следующие паттерны программирования: Factory Design Pattern, Singleton Design Pattern, Bilder Design Pattern, _Data _Access _Object _Pattern.

Тестирование проводилось на узле локальной вычислительной системы, были разработаны дополнительные тесты для проверки логики приложения. Тестирование проводилось согласно сценарию показанному на рисунке 1.