**4 Информационная безопасность приложения**

ФИО

Подпись

Дата

Лист

1

БГТУ 04.00.ПЗ

Разраб.

Провер.

Н. контр.

Утверд.

4 Информационная безопасность приложения

Лит.

Листов

4

74217058, 2020

**4.1 Что такое безопасность?**

В контексте защиты информации «безопасность» относится к надежным системам безопасности, которые помогают защитить информацию от угроз, контролируя, как информация используется, потребляется и предоставляется.

Общие цели информационной безопасности можно разделить на семь категорий:

* конфиденциальность хранит секреты в секрете. Это достигается за счет шифрования данных в состоянии покоя и в пути, а также физического и технического контроля доступа;
* целостность относится к точности и надежности информации и систем. Аппаратные, программные и коммуникационные механизмы должны работать вместе, чтобы правильно поддерживать и обрабатывать данные, а также перемещать данные по назначению без неожиданных изменений;
* доступность гарантирует надежный и своевременный доступ к данным и ресурсам уполномоченным лицам. Оборудование и приложения должны восстанавливаться после сбоев безопасным и быстрым способом, чтобы производительность не оказывала негативного влияния;
* подлинность заключается в надежной проверке подлинности сообщения, получателя или отправителя;
* вменяемость – это способность надежно назначать взаимодействия конкретным случаям или отдельным лицам;
* не респектабельность относится к ассоциированию действий или изменений с уникальным человеком таким образом, что человек не может отрицать действие;
* надежность – это способность предприятия выполнять необходимую функцию в течение определенного периода при определенных условиях.

Некоторые могут сказать, что только первые три категории - конфиденциальность, целостность и доступность - имеют решающее значение для информационной безопасности. Хотя это, безусловно, основные атрибуты, когда речь заходит о информационной безопасности, подлинность, вменяемость, невозможность получения репутации и надежность, возможно, столь же важны, особенно когда вы работаете в системах или храните данные в размещенных системах или в облаке.

4.2 Mule Secure Configuration

Защита свойств конфигурации для *Mule-*приложений включает в себя создание файла свойств безопасной конфигурации, определение безопасных свойств в файле и настройку файла в *Mule*-приложениях с помощью модуля расширения свойств *Mule Secure Configuration*.

Первой задачей в защите свойств конфигурации является создание файла конфигурации *YAML*, в котором определяются свойства. С помощью модуля расширения свойств *Mule Secure Configuration* осуществляется настройка типов файлов *YAML* или свойств.

Далее определяются безопасные свойства в файле. Файл может содержать как зашифрованные, так и незашифрованные значения, которые можно использовать в любом месте *Mule-*приложения.

Настройка файл безопасных свойств осуществляется в *Anypoint Studio* с помощью модуля расширения свойств *Mule Secure Configuration*. На рисунке 4.1 представлена конфигурация модуля расширения свойств *Mule Secure Configuration*.

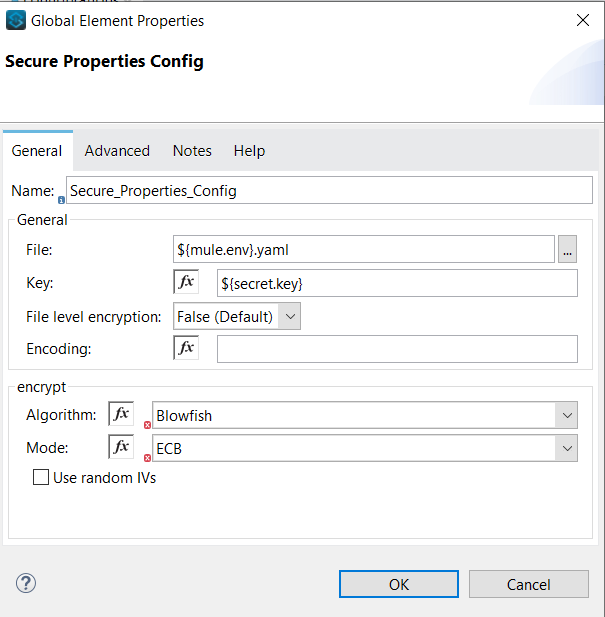


Рисунок 4.1 – Конфигурация модуля *Mule Secure Configuration*

В конфигурации модуля *Mule Secure Configuration* определятся: файл конфигурации, алгоритм шифрования, режим алгоритма шифрования и ключ шифрования. Во время развертывания *Mule*-приложения ключ дешифрования передается в механизм выполнения *Mule* как переменная системной среды *runtime.property*. Это свойство должно быть ключом, используемым для шифрования значений, хранящихся в *YAML* файле.

В моих *Mule-*приложениях все свойства конфигурации помещены в файлы свойств безопасной конфигурации. Файл свойств безопасной конфигурации одного из *Mule-*приложений представлен на рисунке 4.2.

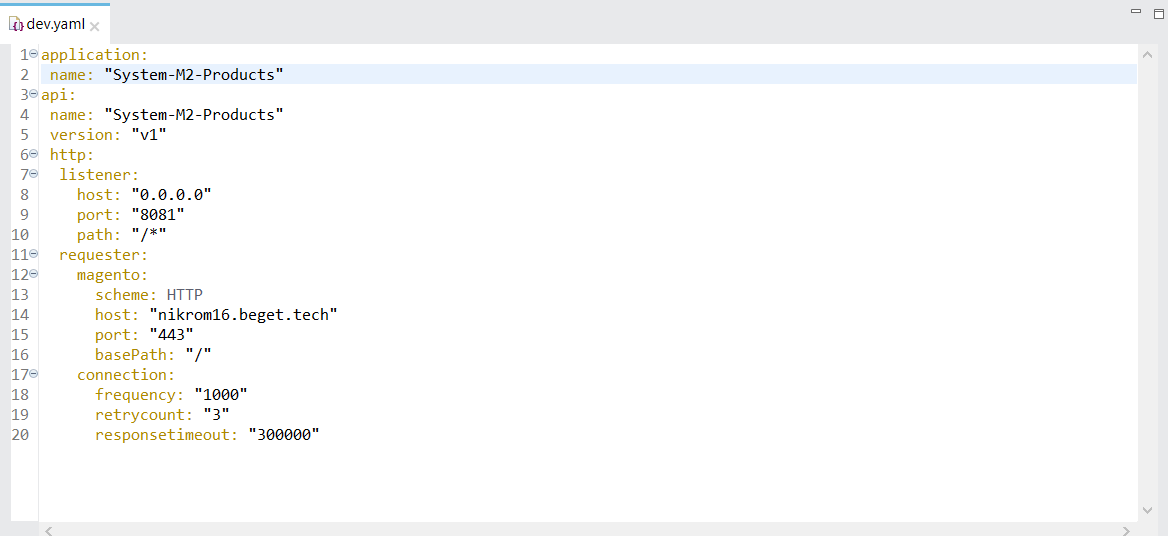


Рисунок 4.2 – Файл свойств безопасной конфигурации

На рисунке 4.3 представлены зашифрованные свойства одного из элементов конфигурации *Mule-*приложения.

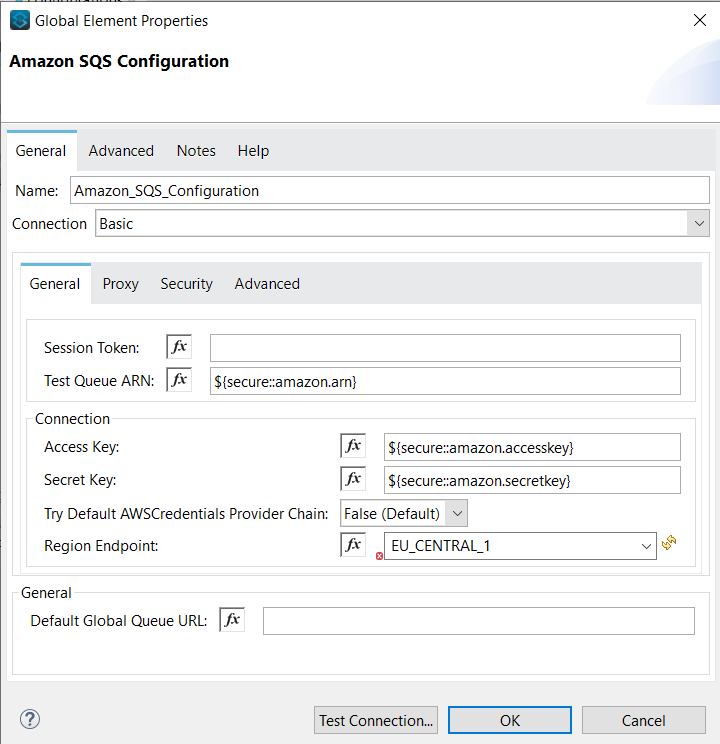


Рисунок 4.3 – Зашифрованные свойства элемента конфигурации

4.3 Протокол HTTPS

*HTTPS* – это расширение протокола *HTTP* для поддержки шифрования в целях повышения безопасности. Данные в протоколе *HTTPS* передаются поверх криптографических протоколов *TLS* или *SSL*. В отличии от *HTTP* с *TCP*-портом 80, для *HTTPS* по умолчанию используется *TCP*-порт 443.

*HTTPS* не является отдельным протоколом. Это обычный *HTTP*, работающий через шифрованные транспортные механизмы *SSL* и *TLS*. Он обеспечивает защиту от атак, основанных на прослушивании сетевого соединения — от *снифферских* атак и атак типа *man-in-the-middle*, при условии, что будут использоваться шифрующие средства и сертификат сервера проверен и ему доверяют.

По умолчанию *HTTPS* *URL* использует 443 *TCP*-порт. Чтобы подготовить веб-сервер для обработки *HTTPS*-соединений, администратор должен получить и установить в систему сертификат открытого и закрытого ключа для этого веб-сервера. В *TLS* используется как асимметричная схема шифрования (для выработки общего секретного ключа), так и симметричная (для обмена данными, зашифрованными общим ключом). Сертификат открытого ключа подтверждает принадлежность данного открытого ключа владельцу сайта. Сертификат открытого ключа и сам открытый ключ посылаются клиенту при установлении соединения; закрытый ключ используется для расшифровки сообщений от клиента.

Существует возможность создать такой сертификат, не обращаясь в центр сертификации. Подписываются такие сертификаты этим же сертификатом и называются самоподписанными.

Чтобы обеспечить конфиденциальность данных в моём дипломном проекте были разработаны *Mule-*приложения, которые предоставляют конечные точки, на основе *HTTPS*. Для этого требовалось:

* Создать файл формата *jks* с помощью утилиты *keytool* в командной строке.
* Добавить сгенерированный файл формата *jks* в папку приложения *Anypoint Studio Mule* в формате *src/main/resources*.
* Настроить *HTTP*-прослушиватель в *Mule*-приложении.

Настройки *HTTP-*прослушивателя представлены на рисунке 4.4.

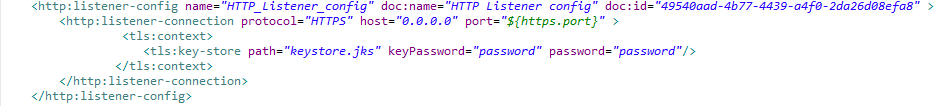


Рисунок 4.4 – Настройки *HTTP*-прослушивателя

4.4 Выводы по разделу

Крайне важно обеспечить безопасность ценной информации, которую компания хранит и делает доступной с помощью программных приложений и веб-служб, от неавторизованных пользователей и злоумышленников. Но также очень важно, чтобы эти защищенные ресурсы, такие как информация о кредитных картах или номера социального страхования, были немедленно доступны авторизованным, законным пользователям и системам для проведения деловых операций.

Чтобы обеспечить безопасный доступ к информации, приложения и службы могут применять различные меры безопасности. Мною были реализованы методы, которые позволяют защищать приложения:

* Защита свойств конфигурации приложения.
* Использование модуля криптографии.
* Использование *HHTPS* протокола.