УДК 681.3:004.056

И. М. Рудяк, магистрант;
А. А. Воронов, доц., канд. техн. наук;
(БГТУ, г. Минск)

ПРИМЕНЕНИЕ ТЕХНОЛОГИИ JAVA ДЛЯ ЦИФРОВОЙ ПОДПИСИ ЭЛЕКТРОННОГО ДОКУМЕНТА

Приложения, которыми будут пользоваться большое количество пользователей, или крупные компании, имеют серьезные требования к безопасности. Пользователи должны быть уверены, что данные, проходящие через приложения, не были изменены, не были украдены, пришли от конкретного человека или дошли до конкретного человека или организации. Если предприятие небольшое, то все эти условия могут быть соблюдены при выполнении общих правил безопасности. Однако, с ростом компании таких мер становится недостаточно. Ситуация становится еще более сложной, когда для передачи данных требуется выход за пределы локальной сети.

Основой для безопасной работы с данными является криптография. Использование криптостойких алгоритмов шифрования, хэширования и электронной подписи обеспечивает безопасность при использовании электронного документооборота.

Java позволяет использовать криптографию для написания приложений поддерживающих безопасность электронного документооборота (ЭДО) как в виде встроенной функциональности, так и в виде сторонних библиотек. Помимо этого, при разработке криптографической части java, была предусмотрена возможность расширения встроенных возможностей в виде использовании криптографических провайдеров. Любой разработчик имеет возможность написать свой криптографический провайдер, в котором будут реализованы механизмы криптографии, и предоставлять этот провайдер конечным пользователям, независимо от самого языка.

Стандартная реализация языка включает в себя провайдер, который предоставляет наиболее распространенные преобразования для шифрования и хеширования, например, DSA, RSA, MD5, SHA и т.п. Для платформы Java предоставляется API. Оно включает в себя криптографию, аутентификацию, инфраструктуру открытых ключей, защищенные коммуникации и контроль доступа. Данное API придерживается следующих принципов:

- независимость реализации;
- совместимость;
- расширяемость алгоритмов.

Независимость имплементации говорит нам, что приложения не должны сами реализовывать криптографические алгоритмы. Java включает в себя набор широко используемых криптографических алгоритмов. Однако, для некоторых приложений требуется специальная реализация криптографических алгоритмов. Java расширяема т.к. поддерживает установку «кастомных» провайдеров.

Криптографический провайдер ссылается на пакет, который предоставляет конкретную реализацию какого-либо криптографического алгоритма. Класс java.security.Provider (JCA) инкапсулирует непосредственно сам провайдер. Он определяет имя провайдера и перечисляет сервисы, которые он реализует. При этом есть возможность настроить несколько провайдеров и определить их приоритет. Когда приложение запрашивает какой-либо сервис, то используется реализация того провайдера, которые имеет наивысший приоритет.

Система JCA позволяет реализовать библиотеки в виде криптопровайдеров java. При этом разработчик, который будет использовать данные криптопровайдеры, сможет легко подключить криптостойкие ЭЦП на базе криптографических хеш-функций семейства SHA-2 и SHA-3. Если требуется интегрировать новый криптопровайдер с ЭЦП в уже существующую систему, то не потребуется никакого изменения кода, кроме замены криптопровайдера. Существуют готовые библиотеки в java для электронной цифровой подписи (ЭЦП) и разработчик может подключать их в своем решении:

- Java включает в себя криптопровайдер SUN с решением для создания ЭЦП;
- Apache Santuario – в составе библиотеки присутствуют алгоритмы DSA, RSA, ECDSA;
- Bouncy Castle – DSA, RSA, ECDSA;
- Cryptix – DSA, ElGamal, RawDSA, RSA;
- FlexiProvider – RSA, DSA, ECDSA, ECNR. ECNR.

ЛИТЕРАТУРА

1. Koret, J. The Antivirus Hacker’s Handbook / J. Koret, E. Bachaalany – John Wiley & Sons, Inc., Indianapolis, Indiana. – 2015. – P. 165-175.

2. Kaspersky Security Bulletin.Overall statistics for 2017 [Electronic resource]. – 2017. – Mode of access : http://media.kaspersky.com/jp/pdf/pr/ Kaspersky_KSB2017_Statistics-PR-1045.pdf. – Date of access : 15.01.2019