System for enhancing the security of private data using public-key cryptography

Danylo Kostyshyn[1], Yevheniya Levus[2]

[1]Software Department, Lviv Polytechnic National University,
UKRAINE, Lviv, S. Bandery street 28a,
E-mail: danylo.kostyshyn@gmail.com

[2]Software Department, Lviv Polytechnic National University,
UKRAINE, Lviv, S. Bandery street 28a,
E-mail: elevus@lp.edu.ua

Abstract* – *This article describes the client-server system for asymmetric-keys encryption of users private correspondence in one of the most popular social network - Facebook. The system uses browser extension as a client, asymmetric-keys algorithm to implement an encryption and the server in turn serves as a public-keys distribution center for users. The system resolves three basic principles of information security - confidentiality, integrity and availability.

Key words – social network, public-key cryptography.

I. Introduction

With a significantly increased amount of information which is transferred via the Internet - fast and simple way of exchanging ideas and information through social networks become a norm in the online world. In regard to this, increases the problem of protection this information, especially when the information is sensitive and confidential. There are many implementations of systems for information protection, all of them can be significantly different from each other in theirs methods and algorithms. However, all of this systems should resolve three main issues: confidentiality - information available only to an authorized user, integrity - protection of information from modification and readiness of information for use in any moment when it is needed. [1].

An efficient way to solve this problem is usage of a cryptographic data protection. [2].

Proposed system is tightly integrated with Facebook and gives opportunity to use cryptographic protection of private correspondence on the basis of an asymmetric-keys algorithm – RSA [3,4]. The system uses a client-server architecture. Client - is an extension for the Google Chrome browser, which is responsible for user interaction, interaction with social network and a server, which in turn, is responsible for the distribution of public keys of users.

II. Existing Solutions

Facebook is one of the few social networks that supports an open protocol for messaging - XMPP. This allows to add support of social network to the most popular desktop clients for messaging, such as Pidgin and Miranda. These applications support additional plug-ins, which allow users to enable encryption. Main disadvantage of this solution - it requires installation of additional, third-party software that may be restricted by the security policy, for example, in a corporate network. Also user will need to spend more time to get used to these new programs. The system, which is proposed in the article, does not forces the user to refuse of his traditional way to interact with the Facebook platform. Also there is a tendency of large companies of refusing to use XMPP Protocol, it was done by Google and VK for stimulating developers to use their own API, which are more powerful than open standards.

III. Main Stages Of The Messaging Algorithm

Overall structure of the messaging algorithm can be split into the following main steps:

1. Installation of extension.

To start working with the system, user need to install an extension for the Google Chrome browser from the Chrome Web Store.

2. Authorization in the system.

Authorization in the system occurs via extension using Facebook account and OAuth 2.0 technology. After successful authentication, user's unique identifier in Facebook will be stored on the system's server.

3. Keys generation.

Extension helps user to generate an RSA key pair - private and public keys, the public key is saved on the system's server and associated with user's Facebook ID, the private key is available only to the user, in case of loss of the private key - decryption of previously received information will be impossible. Import and export of keys are supported. At this stage preliminary preparation for using the system is complete.

4. Messaging.

User opens default Facebook page, selects a contact with whom he would like to start a conversation, there is a button, on the top of the dialog window, that lets you enable encryption. (If both users are registered in the system).

5. Encryption and sending.

Page with the message window is parsed to get raw message text and a Facebook ID of the recipient. After sending the ID of the recipient to the server and successful validation, that the user really is registered in the system, in response, the server send a public key of recipient. Message encrypts with this public key, when the user clicks "Submit" button in the input box, also extension add special signature, which helps identify this message as an encrypted, and after that it is sent via Facebook.

6. Receiving the message.

When a user receives a message, extension tries to find a special signature in a text, which was added in a previous step, and if the signature is found - message is encrypted, extension will try to decrypt it using the private key, then the encrypted text will be replaced with decrypted equivalent in Facebook page.

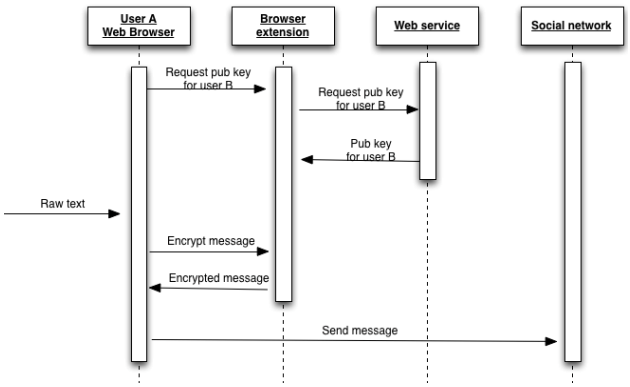

Fig. 1. Send a message to user A

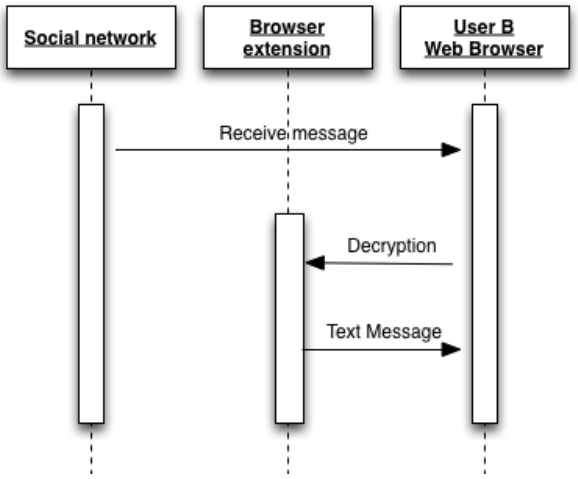

Fig. 2. Receiving the message by the user B

Encryption and sending of a message by the user A and decryption of the message by the user B are depicted on appropriate sequence diagrams. (Fig.1, Fig.2.).

IV. Implementation

This system can be implemented on the client-server architecture, where the client is an extension for Google Chrome browser [5]. Using one of the most popular browser on the market provides independence from the operation system on user side. Encryption and decryption of messages are performed directly by the extension and implemented in JavaScript. For the system like this, the best would be to use asymmetric-keys encryption. In case of using symmetric-key encryption, significant drawback would be a the need to store a key on both sides, and the key may be a target for possible interception through communication channels. [6,7]. Public-key encryption ensures that the user is exempt from transfer of his secret, private key. For these purposes the RSA algorithm works perfectly. Interaction with Facebook and the system's server for obtaining public keys, will also be implemented using the JavaScript language and Facebook Web API.

Server part of the system is a repository of public keys with Facebook IDs, and access interface. For data storage was decided to use SQL solution, widespread MySQL database would be good. External interface for communication and business logic that works with the database, can be implemented using a flexible web framework Django and Python programming language [8].

Conclusion

Organizational and technical approach for solving the problems of confidentiality, integrity and availability of private information can not be applied to users communication in social networks. This article describes a system, which implements a secure, encrypted communication for users, using Facebook social network as a transport for sending messages and asymmetric-keys algorithm. Proposed solution tightly integrates with Facebook social network by using extension for Google Chrome web browser. Extension provides additional controls for enabling encryption directly on a Facebook messages page, so user can continue to use familiar interface, and don't spend time for a learning how to work with entirely new, third party software.

References

[1] V.A. Halatenko, "Bezopasnost' v social'nyh setjah" ["Security in social networks"], Newsland, 2011. [Online]. Available: http://newsland.com/news/detail/id/846433. [Accessed Sep. 15, 2013].

[2] "Kriptografichni metodi zahistu informacii" ["Cryptographic methods of data protection"], Biblionfond, 2010. [Online] Available: http://bibliofond.ru/view.aspx?id=446831. [Accessed Sep. 15, 2013].

[3] C. Paar, J. Pelzl, Understanding Cryptography: A Textbook for Students and Practitioners. New York City, NY: Springer, 2010, pp. 52-63.

[4] T. S. Denis, Cryptography for Developers. Riverport Ln., MO: Syngrees, 2006, pp. 113-120.

[5] Building a Chrome Extension, Chrome Developer, 2013. [Online]. Available: http://developer.chrome.com/extensions. [Accessed Sep. 25, 2013].

[6] N. Ferguson, B. Schneier, T. Kohno, Cryptography Engineering: Design Principles and Practical Applications. Indianapolis, IN: Willey, 2010.

[7] S. Garfinkel, PGP: Pretty Good Privacy. Sebastopol, CA: O'Reilly, 1994, pp. 45-61.

[8] J. Forcier, P. Bissex, W. Chun, Python Web Development with Django. Boston, MA: Addison-Wesley, 2008.