Annales UMCS Informatica AI XII, 4 (2012) 113–122
DOI: 10.2478/v10065-012-0019-x

Annales UMCS
Informatica
Lublin-Polonia
Sectio AI
http://www.annales.umcs.lublin.pl/

Distributed Social Network - data security

Marcin Alan Tunia[1]*

[1] *Institute of Telecommunications of WUT*
Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract – The present day Internet provides a wide range of services users can benefit from. Some of the services require gathering, processing and presenting data that come from many users in order to deliver additional information. The suitable example can be social networking service. The more valuable data it stores and processes, the more profitable it can become. Users' personal data can constitute significant value. One of the issues of social networking is storing and processing data by only one entity. Users cannot choose the most suitable security policy because there is only one provided for certain social network. Being part of it, means accepting the risk of unauthorised data distribution and data leakage because of application vulnerabilities. This paper presents new architecture of social network, which provides mechanisms for dividing data between more than one entity and combining independent data repositories in order to deliver one social network with clearly defined interfaces used to connect new data sources.

1 Introduction

Instant growth in telecommunications sector gives companies possibilities to deliver more complicated and data-oriented services. Fast Internet connections allow service providers to take advantage of data, that is produced by users, such as text content, photos, videos and hyperlinks. The model of services has changed in the last few years. First websites delivered static content originating from the service provider. Now users produce data that is presented to other users on popular websites. Various categories of data can be collected with this new services model, for example:

- personal data (address, phone number, political views etc.);
- personal pictures and videos;
- opinions;

*M.Tunia@stud.elka.pw.edu.pl

- interests;
- connections to other people.

These groups may contain sensitive data, that should be properly processed, according to the law. Users (at least some of them) are also interested in where the data they submitted on the website will be stored, who will have access to it and what it will be used for.

One of the data security problems of social networks is the administrative centralized model of data storage and processing. If one chooses to be part of certain social network, he or she has to entrust only one possible entity with his data. This problem applies to most significant social networks. Maintaining this situation is important for companies because they keep control of all users using their websites and do not have to share benefits. On the other hand, dividing one data storage domain into several domains would increase competitiveness on the social network market and would cause increase in new functions as more people and entities would be involved in development process. Moreover, diffusion of data storage would cause formation of various security and privacy policies as data would be stored and processed by various entities.

The system described in this paper was inspired by the project Diaspora (see [**1**]) and has similar concept, but it is designed with strictly defined interfaces between elements to enable independent parties easy design and development of new components. It also defines general functions for each element of the architecture to enable the designed system to work properly. This paper describes the concept of M.Sc. thesis [**2**].

2 Assumptions

It is assumed that the designed system should provide better security level in comparison with the currently popular social network by data diffusion. Security is taken into account in general architecture to minimize consequences of implementation and design gaps. Architecture forces each component to follow certain rules.

This paper describes 3 layers of specificity: general architecture, sample technologies possible to the used with this project and implementation as a proof of concept.

3 Architecture

The designed social network consists of 2 main parts: domains and main server. Fig. 1 presents the architecture with connections between its elements. The domains communicate with each other to exchange the users' data. The system consists of many domains like e-mail servers network. The main server is a system which enables search user service for domains. Users can use different domains as their interface to the system.

Domains are the systems which store and process the users' data. Internal architecture and used solutions are not specified. Each domain can implement various

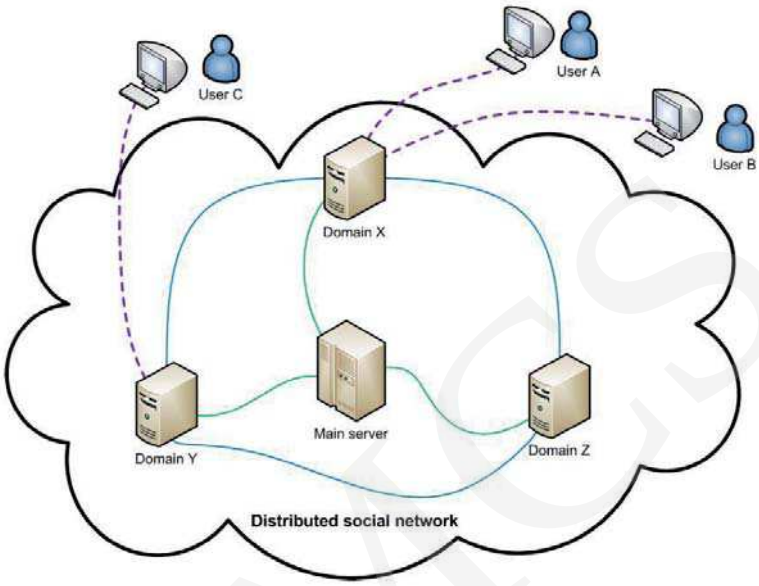

Fig. 1. General architecture.

functions in various ways. It is required that each domain enables other domains and main server to access specified interfaces. Each interface has a defined strict purpose and operations, which can be done with it. The functions performed by the domain are as follows:

- Registration of new users;
- Identification and authentication of registered users;
- Storage of detailed users' data;
- Content processing and presentation;
- Enabling users to define their data view restriction for other users;
- Enabling administrators to define data view restriction for other domains;
- Data distribution according to the defined restrictions by users and domain administrators;
- Exposition of required interfaces;
- Identity confirmation of other domains and main server accessing the interfaces;
- Access control for other domains and main server accessing the interfaces;
- Performing value added functions by combined usage of other architecture's elements and internal data.

The main server does not store or process detailed information about users. It only operates on the data that is required for performing search service. Its way of

implementation is not specified, but like domains it has to enable all domains to access specified interfaces. The functions performed by the main server are as follows:

- Exposition of required interfaces;
- Identity confirmation of domains accessing the interfaces;
- Registration of domains authorised to the user search service;
- Access control for domains accessing the interfaces;
- Collecting and storage of data from the domains required to perform search service;
- Performing user search function.

4 Sample technologies and functions

The previous chapter describes general architecture that can be used with various technologies. To prepare implementation there were chosen specific ones. Each domain should implement the following security mechanisms:

- Domains identification and identity verification with X.509 certificates for both sides of communication;
- Data exchanged with other domains and main server should be done with Secure Socket Layer (SSL);
- Passwords for users' accounts should be stored as one way hash function result, for example SHA-256;
- All data received from users with the graphical user interface, from main server or other domains should be checked in terms of existence of malicious code;
- User identification and authorization should be handled with at least login and password.

Main server should also communicate with domains using SSL, verify certificates and hold its own X.509 certificate. It should also check incoming data to avoid execution of malicious code.

5 Interfaces

Interfaces described in this paper are designed for web services (see [**3**]), so there is definition for input and output XML file structure. Web services available to invoke in domains are as follows:

- getUserData – is used to obtain detailed data about the user registered in the domain it is invoked in;
- sendMessage – is used to send a message to the user registered in the domain it is invoked in;

- sendInvitation – is used to send a friendship invitation to the user registered in the domain it is invoked in. It can be treated as a request for adding a certain user from the other domain to the user's friends list;
- getUserStatuses – is used to obtain statuses (short text messages that are addressed to a certain group of users) of the user registered in the domain the web service is invoked in;
- invitationResponse – is used to send acceptance or rejection of the previously sent friendship invitation originated from the domain the web service is invoked in;
- deleteConnection – is used to send information about removing a certain user (registered in the domain the web service is invoked in) from the other user's (registered in foreign domain) friends list;
- getAllUsersToSearch – is used to obtain general information about the users (registered in the domain the web service is invoked in) that should be visible in the search service;
- getAvailableWebServices – is used to obtain a list of web services available in the domain it is invoked in.

The main server hosts the following web services:

- searchUser – is used to obtain general data about users and their domains. Results are determined by regular expressions for name, surname, username;
- updateUser – is used to register or update the user's data (username, domain name, name, surname) in the main server database, which is used for search service;
- getAvailableWebServices – is used to obtain a list of web services available in the main server.

Tables 1-10 show the detailed description of input and output data for web services described in this section. For each web service there is always defined the "error code" field in output parameters which indicates that operation is successful or specify the cause of failure. In input data there is no domain specification because the domain for requesting user is determined with certificate, which contains the full domain name that is globally unique. For the users who are the object of the query we also do not need domain name, because it is assumed that the user is in the domain in which the web service is invoked.

6 Implementation

The designed system architecture was used to prepare implementation as a proof of concept. Domain and main server applications were developed using the PHP language. Domain applications can be copied to simulate multiple domains in the system. Each element of the architecture (domains and main server) should have its own self signed X.509 certificate with the unique name in the organizational unit (OU) field. All applications were placed on the same Apache server so they use the same certificate. To

Table 1. getUserData web service.

	getUserData
Input parameters	• username for the user that request for data • username for the user whose data have to be delivered
Output parameters	• name • surname • sex • place of residence • "about me" information • interests • education • professional work • e-mail address • error code

Table 2. sendMessage web service.

	sendMessage
Input parameters	• username of message sender • username of message receiver • message title • message content
Output parameters	• error code

Table 3. sendInvitation web service.

	sendInvitation
Input parameters	• username of invitation sender • username of invitation receiver
Output parameters	• error code

identify a domain there was added a new field in each web service input indicating the unique invoking entity name. Each domain and main server have their own database so they do not have access to each other's data. Communication between the architecture elements as well as between the client and the domains is done using the SSL protocol. Web services were developed using the PHP PEAR SOAP extension.

On the production server there were deployed three domain applications and one main server application. Each has a graphical installation guide. Domains needed to be registered in the main server before the first use. In the domains after successful

Table 4. getUserStatuses web service.

	getUserStatuses
Input parameters	• username of the user requesting statuses • username of the user whose statuses should be retrieved • maximum number of statuses in response (used to avoid processing of too big XML files)
Output parameters	• list of values: ○ list of statuses ○ list of published dates in the same order as the list of statuses (the first date relates to the first status on the list, the second date relates to the second status, etc.) • flag determining if the response includes all statuses possible to obtain with the current query • error code

Table 5. invitationResponse web service.

	invitationResponse
Input parameters	• username of invitation sender • username of invitation receiver • response status - accept/reject the invitation
Output parameters	• error code

Table 6. deleteConnection web service.

	deleteConnection
Input parameters	• username of the user who deletes the connection • username of the user with whom the connection is deleted
Output parameters	• error code

installations there should be created the first user who will be a site administrator. Additional administrators can be nominated by modifying user's profile in the database.

7 Maintenance

After successful production deployment all applications in the system should be monitored and patched to ensure high security level. The following actions are recommended:

Table 7. getAllUsersToSearch web service.

	getAllUsersToSearch
Input parameters	• maximum number of users in response
Output parameters	• list of values (it is assumed that the data with the same indices from the lists above relate to the same user): ○ list of usernames ○ list of names ○ list of surnames • flag determining if the response includes all users possible to obtain with the current query • error code

Table 8. getAvailableWebServices web service.

	getAvailableWebServices
Input parameters	None
Output parameters	• list of web services names (e.g. getUserData, sendMessage, getUserStatuses) • error code

Table 9. updateUser web service.

	updateUser
Input parameters	• username • name • surname
Output parameters	• error code

- Automatic tests applications (e.g. Selenium [**4**]) should verify regularly main functions of domains and main server according to the previously specified test scenarios;
- There should be done penetration tests at least after implementing application code changes;
- Constant logs analysis in order to find suspicious web service queries and failed login attempts;
- Application of performance control to avoid server overload and a lack of database or system disk space;
- Error reporting channel for users to provide application problem reporting;
- Discovered problems patching in the shortest time possible after bug detection.

Table 10. searchUser web service.

	searchUser
Input parameters	• regular expression for the user's name • regular expression for the user's surname • regular expression for the user's username • maximum number of users in response
Output parameters	• list of values (it is assumed that the data with the same indices from the lists above relate to the same user): ○ list of usernames ○ list of names ○ list of surnames ○ list of domain names ○ list of domain addresses • flag determining if the response includes all users possible to obtain with the current query • error code

8 Security benefits

New architecture with cloud of domains working together as one social network can improve users' data security. The user is now responsible for his or her data by choosing a domain to sign in. Various domains can provide various levels of data security. Each domain should declare the way it stores and processes data in privacy policy, which a new user has to accept. In case of security problems of one domain not all data are exposed. In the worst scenario of data leakage in one domain an attacker can obtain data stored in that domain and data possible to download by the compromised domain from other domains. Restricted data distribution outside domains helps to minimize consequences of security problems that take place in production environments (e.g. [**5, 6**]).

It is worth analysing the influence of the architecture on resistance against popular attacks on web applications. Unauthorised data processing by entity storing data is very hard to perform on a global scale because of diffused data repositories. Vulnerability to web application attacks such as SQL injection or Cross Site Scripting (XSS) depends on implementation of certain domain. Size of damage is limited to part of the social network, because of existence of several domains. Successful attacks like denial of service (DoS) or distributed denial of service (DDos) can cause inaccessibility of only one domain. The whole social network will be still accessible. Web services can be the objects of DoS or DDos attacks, so they should be secured at least by a limiting number of simultaneously processed requests to the defined value in order to prevent overloading server's capacity. Giving priorities for connections from known addresses would be also a good practice. Other implementation errors and security mechanisms to prevent for example user account hijacking are dependent on the domain and what security policy it delivers. If one cannot find suitable policy, there is possibility to develop a new domain with the desired functions.

It is worth mentioning that various domains can be physically placed in various geographical regions so that they are working under different laws. The users would have a wide range of available security policies fitted to be consistent with the local law, which regulates also security means, for example protocols and cryptographic strength of ciphers available for the use in civil applications.

9 Conclusions and future research

Distributed social network may cause competitiveness between vendors and would result in various security policies and functions. In this new model of social networking users can choose the most suitable vendor by choosing a certain domain. Architecture presented in this paper would have applications in the business area (intranet corporation portal divided into department domains), academic area (exchange of information in a large project group or intra-university portal) and using the Internet network (worldwide portal with multiple geographically separated domains).

Only one main server presented in the architecture may limit scalability of the solution, but for larger applications it should be replaced with the network of main servers working together to deliver the search function.

A list of available web services can be extended by new ones in order to support new functions and make the whole network more flexible. Various domains may implement various web services, but there must be a mandatory group of web services delivered by all domains (e.g. those described in this paper).

References

[1] The Diaspora Project (2012-05-31); http://diasporaproject.org/
[2] Tunia M. A., M.Sc. thesis, Distributed social network - data security, Warsaw University of Technology, Warsaw, Poland (2012).
[3] W3C, Web Services Architecture (2012-05-31); http://www.w3.org/TR/ws-arch/
[4] Selenium IDE - Selenium Documentation (2012-05-31); http://seleniumhq.org/docs/02_selenium_ide.html#introduction
[5] Symantec - Nishant Doshi. Facebook Applications Accidentally Leaking Access to Third Parties - Updated (2012-05-31); http://www.symantec.com/connect/blogs/facebook-applications-accidentally-leaking-access-third-parties
[6] Symantec - Candid Wueest. Persistent XSS Vulnerability in Facebook (2012-05-31); http://www.symantec.com/connect/blogs/persistent-xss-vulnerability-facebook