УДК 519.718.7:681.3.069

**С. С. Сурков,**
**А. Н. Мартынюк, И. Г. Милейко**, кандидаты техн. наук

## МОДИФИКАЦИЯ ОТКРЫТОГО ПРОТОКОЛА АВТОРИЗАЦИИ ДЛЯ ВЕРИФИКАЦИИ ЗАПРОСОВ

***Аннотация***. *Авторизация пользователя в социальных сетях порождает проблему защиты информации от сторонних приложений. Для ее решения обычно применяется протокол OAuth, вместе с тем, разработка интернет-приложений требует также и обеспечения защиты данных, что отсутствует в OAuth версии 1.0, в частности, для запросов, отличных от URL-encoded. В данной статье предлагается модификация протокола OAuth, классов и алгоритма генерации подписи, обеспечивающая поддержку подписи любых запросов.*

***Ключевые слова:*** *сервер, Java, OAuth, REST API, смартфон, iOS, Android*

**S. Surkov,**
**A. Martynyuk**, PhD., **I. Mileiko**, PhD.

## MODIFICATION OF OPEN AUTHORIZATION PROTOCOL FOR VERIFICATION OF REQUEST

***Abstract.*** *User's authorization social networks create a problem of protection of information from third-party applications. To solve it is commonly used protocol OAuth, however, the development of Internet applications, and also requires the protection of data that is not in the OAuth 1.0, in particular for requests other than the URL-encoded. In this paper we propose a modification of the protocol OAuth, classes and signature generation algorithm, which provide support for the signature of any requests.*

***Keywords:*** *server, Java, OAuth, REST API, Smartphone, iOS, android*

**С. С. Сурков,**
**О. М. Мартинюк**, **І. Г. Мілейко**, кандидати техн. наук

## МОДІФІКАЦІЯ ВІДКРІТОГО ПРОТОКОЛУ АВТОРИЗАЦІЇ ДЛЯ ВЕРИФІКАЦІЇ -ЗАПИТІВ

***Анотація.*** *Авторизація користувача в соціальних мережах породжує проблему захисту інформації від сторонніх додатків. Для її рішення зазвичай застосовується протокол OAuth, разом з тим, розробка інтернет-додатків вимагає також і забезпечення захисту даних, що відсутня в OAuth версії 1.0, зокрема, для запитів, відмінних від URL-encoded. У даній статті пропонується модифікація протоколу OAuth класів і алгоритму генерації підпису, що забезпечує підтримку підписи будь-яких запитів.*

***Ключові слова:*** *сервер, Java, OAuth, REST API, смартфон, iOS, android*

**Введение**

Безопасность данных необходимы для выполнения операций авторизации, чтобы данные были защищены от несанкционированного доступа. В случае ограниченного предоставления такового обычно используются открытые протоколы авторизации.

Самым распространенным открытым протоколом авторизации является OAuth, позволяющий предоставить третьей стороне ограниченный доступ к защищенным ресурсам без необходимости передавать ей (третьей стороне) логин и пароль. Например, пользователь, который хочет предоставить сервису социальной сети доступ к книге контактов своего почтового аккаунта, не должен сообщать сети свой пароль от почты. Вместо этого он проходит авторизацию непосредственно в почтовом сервисе и с разрешения владельца или администратора сервиса получает полномочия доступа к адресной книге.

В работе [1] была анонсирована серверная, легко модифицируемая и интегрируемая библиотека OAuth, поддерживающая протоколы OAuth 1.0a и xAuth.



Однако при исследовании протокола OAuth был выявлен недостаток, заключающийся в поддержке подписи только для URL-encoded запросов.

**Цель работы**

Целью работы является концептуальная и алгоритмическая модель верификации тела запроса с любым контентом за счет модификации протокола OAuth 1.0a, расширяющей созданную ранее OAuth библиотеку[1] для фреймворка JAX-RS 2.0 [2 – 3] без потери возможности поддержки стандартного OAuth.

**Особенности реализации подписи запросов OAuth-сервера**

Важной частью OAuth 1.0a является реализация подписи запроса. Согласно спецификации OAuth [1] подпись может быть реализована тремя алгоритмами: RSA SHA1, HMAC SHA1 и обычным текстом. Последний вариант не является безопасным, так как ключи могут быть перехвачены злоумышленником. Использование SSL 2.0 шифрования и проверки сертификата сервера на стороне клиента является наилучшим, так как меньше других вариантов потребляет вычислительные ресурсы.

Диаграмма UML реализации подписи OAuth представлена на рис. 1.

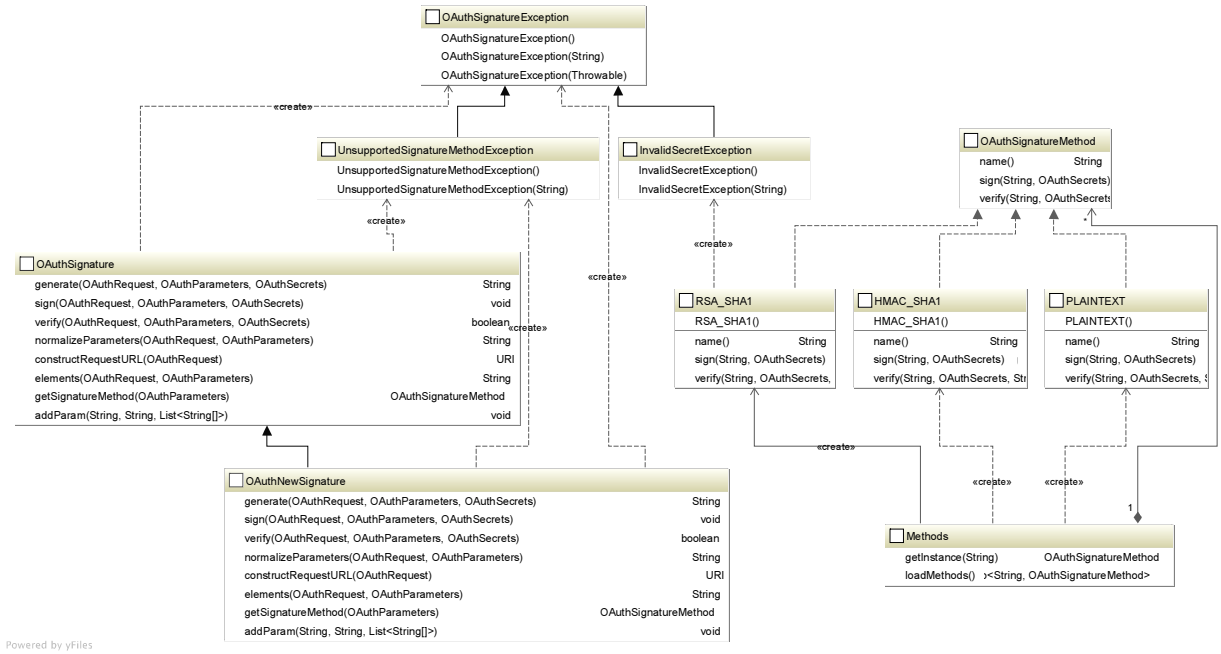


Рис. 1. Модифицированная UML диаграмма классов

В диаграмме главным классом является OAuthSignature, который позволяет подписывать и верифицировать OAuth запросы [1; 4]. При его применении использованы три класса реализации алгоритмов, реализующие методы RSA SHA1, HMAC SHA1 и обычного текста и имплементирующие интерфейс OAuthSignatureMethod. Сам интерфейс нуждается в методах name, sign, verify. Объекты представленных классов единично создаются в фабрике Methods.

Особенность UML диаграммы – добавление класса OAuthNewSignature, являющегося наследником OAuthSignature и верифицирующего подпись модифицированного OAuth/XAuth протокола.

**Алгоритм генерации подписи OAuth**

Для генерации подписи OAuth [5] запросов формируется строка, которая генерируется из параметров заголовка Authorization и URL-encoded параметров запроса.

Также в регулирующей строке к этим параметрам добавляется Consumer Secret.

Для регулирующей строки генерируется SHA-1 хэш, который используется для верификации запроса.

Оригинальный алгоритм генерации подписи показан на рис. 2.

**Ограничения протокола OAuth**

Ограничением протокола OAuth является невозможность проверки целостности POST запросов, формат данных entity у которых отличен от URL-encoded.

При использовании других форматов в теле запроса верификация не выполняется. Данная уязвимость может быть использована злоумышленником во время атаки “man-in-the-middle”, что возможно, если данные не шифруются средствами SSL.

Данная уязвимость присутствует и в протоколе OAuth 2.0, выдающем только временные токены. Здесь также может быть произведена атака “man-in-the-middle”, если в WEB-библиотеках клиента допускается установка сторонних SSL сертификатов [6 – 7].

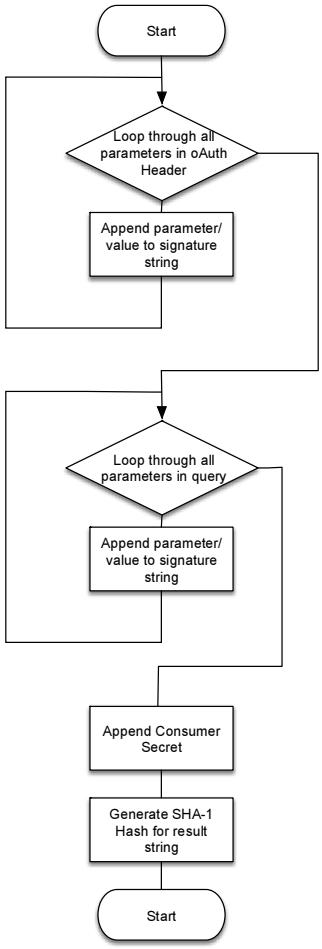


Рис. 2. Оригинальный алгоритм генерации подписи OAuth

В данной работе предложена модификация протокола OAuth [8], которая предлагает проверку контента HTTP [9] запроса.

**Модификация процедуры генерации подписи**

Изменение протокола заключается в изменении строки, использующейся для верификации. В начальном варианте добавляются URL-encoded entity, в измененном методе используется вектор URL-encoded параметров и все данные entity [10].

Модифицированный алгоритм генерации подписи показан на рис. 3.

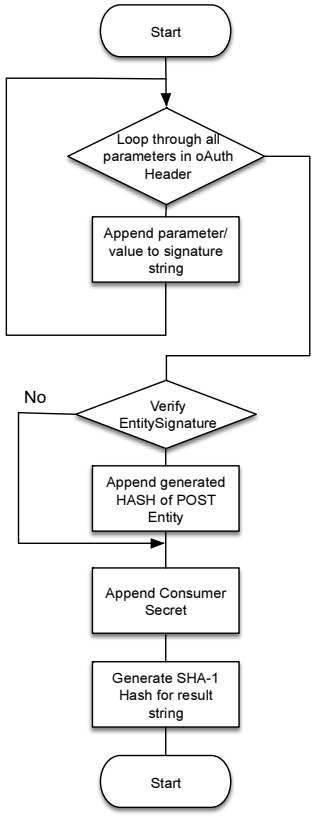


Рис. 3. Модифицированный алгоритм генерации подписи

Данное изменение расширяет возможности применения протокола, но верифицирует весь контент, даже для тех запросов, где это не обязательно.

Для управления подписью тела запроса вводится заголовок, в котором определяется необходимость подписи запроса и передается булево значение, используемое для согласования верификации клиента и сервера, что требуется верификацией тела запроса.

**Сравнение времени верификации при разных длинах запроса**

Для установления рекомендуемой длины тела запроса, при которой использование верификации является допустимым, был разработан тест, который для 40 потоков рассчитывает SHA-1 хэш на случайных фрагментах.

При определении зависимости среднего времени расчета хэша от длины строки были использованы следующие длины строк: 512Kb, 1Mb, 2Mb, 4 Mb, 8 Mb.

В ходе экспериментального исследования каждую секунду обновлялось среднее время подсчета хэш в миллисекундах для 40 потоков.

Результаты исследования показаны на рис. 4.

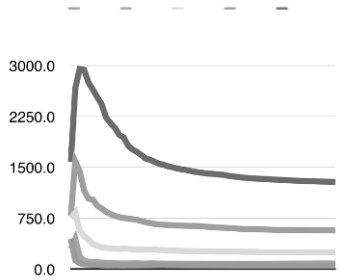


Рис. 4. Изменение среднего времени для размеров хэшируемой строки 0,5; 1; 2; 4; 8 Mb

Для изучения зависимости среднего времени обработки от размера хэш была построена диаграмма с конечным средним временем расчета хэш для разных размеров строк (рис. 5).

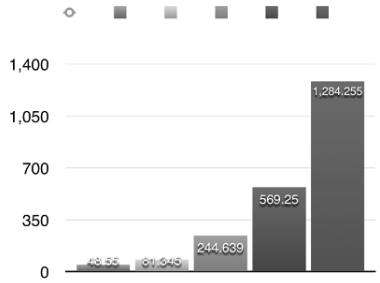


Рис. 5. Среднее время расчета хешей для разных размеров строк 0,5; 1; 2; 4; 8 Mb

Полученные результаты показывают экспоненциальную зависимость $2^x$, что говорит об удорожании проверки содержимого запроса на каждый мегабайт, поэтому рекомендуется в запросах, использующих верификацию, хранить важные данные.

Также можно отметить, что приведенное ограничение распространяется и на URL-encoded запросы, но на практике для них оно менее существенно, поскольку в URL-encoded обычно не передаются большие объемы данных.

**Выводы**

В работе представлена модификация протокола OAuth, позволяющая использовать его для защиты тел запросов любых форматов. Модификация значительно расширило область применения протокола OAuth.

Эксперимент с одновременным выполнением 40 запросов с большим телом показал, что с полной верификацией тела запроса в экспоненциальной зависимости $2^x$ возрастает время расчета хэша.

Использование данной модификации накладывает требование на средний размер запроса, при большом запросе использовать альтернативные решения.

Список использованной литературы

1. Сурков С. С. Авторизация автомобильного компьютера без поддержки браузера посредствам Bluetooth / С. С. Сурков, А. Н. Мартынюк // Холодильна техніка і технологія. – 2015. – № 2. – С. 65 – 71.

2. Richardson Leonard, and Ruby Sam, (2007), RESTful Web Services Web Services for the Real World, *O'Reilly Media May 2007,* 454 p. Available at: Url: http://shop.oreilly.com/product/9780596529260.do (Accessed 24.05.2007).

3. Masse Mark, (2013), REST API Design Rulebook, *O'Reilly Media*, 112 p., Available at: Url: http://shop.oreilly.com/product/0636920021575.do (Accessed 12.10.2011).

4. Hammer-Lahav E. (ed.), (2010), The OAuth 1.0 Protocol, *IETF RFC 5849 (Informational)* April 2010, [Electronic Resource] Available at: Url: http://tools.ietf.org/html/rfc5849 (Accessed 19.04.2010).

5. Basney Jim, and Jeff Gaynor, (2014), National an OAuth Service for Issuing Certificates to Science Gateways for TeraGrid Users, *Center for Supercomputing Applications, University of Illinois at Urbana-Champaign 1205 West Clark Street,* Urbana, Illinois 61801, July 2014. Available at:
Url: http://www.slideserve.com/cleta/an-oauth-service-for-issuing-certificates-to-science-gateways-for-teragrid-users (Accessed 21.07.2014).

6. Dierks T., and. Rescorla E., (eds.), (2008), The Transport Layer Security (TLS) Protocol, *IETF RFC 5246 (Standards Track)*, August 2008, Available at: Url: http://tools.ietf.org/html/rfc5246 (Accessed 25.08.2008).

7. Rescorla E., (ed.), (2000), HTTP over TLS, *IETF RFC 2818* (Informational), May 2000. Available at: Url: http://tools.ietf.org/html/rfc2818 (Accessed 29.05. 2000).

8. LeBlanc J., (2011), Programming Social Applications: Building Viral Experiences with OpenSocial, OAuth, OpenID, [Electronic Resource], Available at: Url: http://www.torrentino.net/torrent/282288 (Accessed 19.12.2011).

9. Fielding R., and Reaches J., (2014), Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing, *IETF RFC 7230* June 2014, [Electronic Resource] Available at: Url: https://tools.ietf.org/html/rfc7230 (Accessed 16.06.2014).

10. Bray T. (ed.), (2014), The JavaScript Object Notation (JSON) Data Interchange Format, Ed *IETF RFC 7159*, March 2014, Available at: Url: http://tools.ietf.org/html/rfc7159
(Accessed 26.03. 2014).

Получено 28.05.2015

References

1. Surkov S.S., and Martynyuk A.N. Avtorizatsiya avtomobilnogo kompyutera bez poddergki brauzera posredstvom Bluetooth [Authorization for Automobile head unit Without Browser Support with Mobile Devices through Bluetooth], (2015), *Refrigeration Engineering and Technology,* No. 2, pp. 65 – 71 [In Russian].

2. Richardson Leonard, and Ruby Sam, (2007), RESTful Web Services Web Services for the Real World, *O'Reilly Media May 2007,* 454 p. Available at: Url: http://shop.oreilly.com/product/9780596529260.do (Accessed 24.05.2007).

3. Masse Mark, (2013), REST API Design Rulebook, *O'Reilly Media,* 112 p., Available at: Url: http://shop.oreilly.com/product/0636920021575.do
(Accessed 12.10.2011).

4. Hammer-Lahav E. (ed.), (2010), The OAuth 1.0 Protocol, *IETF RFC 5849 (Informational)* April 2010, [Electronic Resource] Available at: Url: http://tools.ietf.org/html/rfc5849 (Accessed 19.04.2010).

5. Basney Jim, and Jeff Gaynor, (2014), National an OAuth Service for Issuing Certificates to Science Gateways for TeraGrid Users, *Center for Supercomputing Applications University of Illinois at Urbana-Champaign 1205 West Clark Street*, Urbana, Illinois 61801, July 2014, Available at:
Url: http://www.slideserve.com/cleta/an-oauth-service-for-issuing-certificates-to-science-gateways-for-teragrid-users (Accessed 21.07.2014).

6. Dierks T., and E. Rescorla, (eds.), (2008), The Transport Layer Security (TLS) Protocol, *IETF RFC 5246 (Standards Track)*, August 2008. Available at: Url: http://tools.ietf.org/html/rfc5246 (Accessed 25.08.2008).

7. Rescorla E., (ed.), (2000), HTTP over TLS, *IETF RFC 2818* (Informational), May 2000, Available at: Url: http://tools.ietf.org/html/rfc2818 (Accessed 29.05.2000).

8. LeBlanc J., (2011), Programming Social Applications: Building Viral Experiences with OpenSocial, OAuth, OpenID, [Electronic Resource], Available at: Url: http://www.torrentino.net/torrent/282288 (Accessed 19.12.2011).

9. Fielding R., and Reaches J, (2014), Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing, *IETF RFC 7230* June 2014. [Electronic Resource] Available at: Url: https://tools.ietf.org/html/rfc7230 (Accessed 16.06.2014).

10. Bray T. (ed.), (2014), The JavaScript Object Notation (JSON) Data Interchange Format, Ed *IETF RFC 7159*, March 2014, Available at: Url: http://tools.ietf.org/html/rfc7159 (Accessed 26.03.2014).

Сурков
Сергей Сергеевич;
аспирант каф. компьютерных интеллектуальных систем и сетей Одесского нац. политехн. ун-та,
тел.: +38(091) 916-40-91.
E-mail: k1x0r@ukr.net

Мартынюк
Александр Николаевич,
канд. техн. наук, доц. каф. компьютерных интеллектуальных систем и сетей Одесского нац. политехн. ун-та,
тел.: +38(067) 489-81-69.
E-mail: anmartynyuk@ukr.net

Милейко
Игорь Генрихович,
канд. техн. наук, доц. каф. компьютерных интеллектуальных систем и сетей Одесского нац. политехн. ун-та,
тел.: +38(067) 489-81-69.
E-mail: anmartynyuk@ukr.net