Callback-уведомления

Альфа Банк

Exported on 11/11/2024

Table of Contents

1 Общие сведения

1.1 Типы операций, на которые могут быть получены уведомления

Продавец может получать от платёжного шлюза callback-уведомления (уведомления обратного вызова) об операциях с заказами, перечисленными в таблице ниже.

Операция	**Система оплаты**
Удержание (холдирование) средств	Только двухстадийная система оплаты
Списание средств	Одностадийная и двухстадийная системы оплаты
Отмена перевода средств	
Возврат средств	
Сохранение карты (т.е. создание связки)	
Активация/деактивация существующей связки	
Отклонение платежа из-за истечения времени ожидания	
Отклонение платежа физической картой	

Тип операции передается в параметре `operation` уведомления обратного вызова (см. Формат URL callback-уведомлений (see page 6)).

1.2 Типы уведомлений

Уведомления могут быть двух типов (см. таблицу ниже).

Тип уведомления	Описание
Уведомления без контрольной суммы	Такие уведомления содержат только сведения о заказе - потенциально продавец рискует принять уведомление, отправленное злоумышленником, за подлинное.
Уведомления с контрольной суммой	Такие уведомления, помимо сведений о заказе, содержат аутентификационный код. Аутентификационный код представляет собой контрольную сумму сведений о заказе. Эта контрольная сумма позволяет убедиться, что callback-уведомление действительно было отправлено платёжным шлюзом. Существует два способа реализации callback-уведомлений с контрольной суммой: • с помощью симметричной криптографии - для формирования контрольной суммы на стороне шлюза и для её проверки на стороне продавца используется один и тот же (симметричный) криптографический ключ; • с помощью асимметричной криптографии - для формирования контрольной суммы на стороне платёжного шлюза используется закрытый ключ, известный только шлюзу, а для подтверждения контрольной суммы используется связанный с закрытым ключом открытый ключ, который известен продавцам и может распространяться свободно. Для большей безопасности рекомендуется использовать способ, в котором контрольная сумма формируется с помощью асимметричной криптографии. Чтобы включить возможность получения уведомлений с контрольной суммой и получить закрытый ключ, обратитесь в службу технической поддержки.

1.3 Требования к SSL-сертификатам сайта магазина

Если для доступа к магазину, который работает с callback-уведомлениями, используется HTTPS-соединение, сертификат сайта, на котором расположен этот магазин, должен соответствовать следующим требованиям (см. таблицу ниже).

Требование	**Описание**
Длина и тип ключа сертификата.	Ключ RSA не менее 2048 бит.
Алгоритм подписи.	Не ниже SHA-256.
Поддерживаемые центры сертификации.	Ниже представлены примеры организаций, осуществляющих регистрацию сертификатов: • Thawte Consulting cc – https://www.thawte.com/; • VeriSign – https://www.verisign.com/; • DigiCert Inc – https://www.digicert.com/; • COMODO CA Limited – https://www.comodo.com/; • GeoTrust Inc. – https://www.geotrust.com/; • GlobalSign – https://www.globalsign.com/; • Trustis Limited – http://www.trustis.com/; • UniTrust – http://www.unitrust.co.uk/. Также существует возможность оформления сертификатов через поставщиков в России: • RU-CENTER – http://ssl.ru/; • REG.RU – https://www.reg.ru/ssl-certificate/; • MySSL – https://myssl.ru/. Не допускается использование самоподписанных сертификатов. Сертификат должен быть подписан доверенным центром сертификации (см. выше).

2 Формат URL callback-уведомлений

Уведомление без контрольной суммы

```
{merchant-url}?mdOrder={mdOrder}&orderNumber={orderNumber}&
operation={operation}&status={status}&callbackCreationDate={callbackCreationDate}
```

Уведомление с контрольной суммой

```
{merchant-url}?mdOrder={mdOrder}&orderNumber={orderNumber}&
checksum={checksum}&operation={operation}&status={status}
&callbackCreationDate={callbackCreationDate}
```

Передаваемые параметры представлены в таблице ниже.

> В таблице приведены базовые параметры. В личном кабинете также существует возможность указать ряд дополнительных параметров, которые будут передаваться в уведомлениях. Подробнее см. инструкцию администратора по работе с консолью.

Имя параметра	Описание
mdOrder	Уникальный номер заказа в платёжной системе.
orderNumber	Уникальный номер (идентификатор) заказа в системе магазина.
checksum	Аутентификационный код, или контрольная сумма, полученная из набора параметров.
callbackCreationDate	Время создания запроса уведомления обратного вызова

Имя параметра	Описание
operation	Тип операции, о которой пришло уведомление: • `approved` - операция удержания (холдирования) суммы; • `deposited` - операция завершения; • `reversed` - платеж был отменен; • `refunded` - деньги за заказ возвращены; • `bindingCreated` - карта плательщика сохранена (связка создана); • `bindingActivityChanged` - существующая связка была активирована/деактивирована; • `declinedByTimeout` - платеж был отклонен из-за истечения времени ожидания; • `declinedCardpresent` - отклонена транзакция с предъявлением карты (оплата физической картой).
status	Индикатор успешности операции, указанной в параметре `operation`: • `1` - операция прошла успешно; • `0` - операция завершилась ошибкой.
bindingId	Идентификатор связки.
clientId	Идентификатор покупателя в системе магазина (например, логин).
enabled	Указывает на то, активирована ли связка: • `true` - связка активирована; • `false` - связка деактивирована.
operationRefundedAmount	Параметр возвращает сумму частичного возврата в минимальных значениях
operationRefundedAmountFormatted	Параметр возвращает сумму частичного возврата в минимальных значениях и отформатированную в соответствии с валютой

3 Примеры callback-уведомлений

3.1 Проведение платежа

Пример URL уведомления без контрольной суммы

```
https://myshop.ru/callback/?mdOrder=1234567890-098776-234-522&
orderNumber=0987&operation=deposited&
callbackCreationDate=Mon Jan 31 21:46:52 MSK 2022&status=0
```

Пример URL уведомления с контрольной суммой

```
https://myshop.ru/callback/?mdOrder=1234567890-098776-234-522&
orderNumber=0987&checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A25C14FD813888E231A99E12
20AB3&
operation=deposited&callbackCreationDate=Mon Jan 31 21:46:52 MSK 2022&status=0
```

3.2 Проведение платежа с созданием связки

Пример URL уведомления без контрольной суммы

```
https://myshop.ru/callback/?mdOrder=
1234567890-098776-234-522&orderNumbe
r=0987&operation=deposited&status=0&
jsonParams:{"bindingId":"37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a","clientId":"1","enable
d":"true"}
```

Пример URL уведомления с контрольной суммой

```
 https://myshop.ru/callback/?mdOrder=
1234567890-098776-234-522&orderNumbe
r=0987&checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A
25C14FD813888E231A99E1220AB3&operation=deposited&status=0&
jsonParams:{"bindingId":"37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a","clientId":"1","enable
d":"true"}
```

3.3 Создание связки без проведения платежа

Пример URL уведомления без контрольной суммы

```
https://myshop.ru/callback/?
bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=1&enabled=true
```

Пример URL уведомления с контрольной суммой

```
https://myshop.ru/callback/?checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A
25C14FD813888E231A99E1220AB3&bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=
1&enabled=true
```

3.4 Деактивация связки

Пример URL уведомления без контрольной суммы

```
https://myshop.ru/callback/?
bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=1&enabled=false
```

Пример URL уведомления с контрольной суммой

```
https://myshop.ru/callback/?checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A
25C14FD813888E231A99E1220AB3&bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=
1&enabled=true
```

3.5 Активация ранее деактивированной связки

Пример URL уведомления без контрольной суммы

```
https://myshop.ru/callback/?
bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=1&enabled=true
```

Пример URL уведомления с контрольной суммой

```
https://myshop.ru/callback/?&checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A
25C14FD813888E231A99E1220AB3&bindingId=37e2a02e-9f7b-4335-9e45-7a6a1ec2c95a&clientId=
1&enabled=true
```

4 Алгоритм обработки callback-уведомлений

В таблице ниже представлен алгоритм обработки callback-уведомлений в зависимости от типа таких уведомлений.

Уведомление без контрольной суммы	1. Платёжный шлюз отправляет на сервер продавца HTTP-запрос `GET` следующего вида. `https://myshop.ru/callback/?mdOrder=1234567890-098776-234-522&orderNumber=0987&operation=deposited&status=0` 2. Сервер отправляет в платёжный шлюз HTTP-код `200 OK`.

Уведомление с контрольной суммой	1. Платёжный шлюз отправляет на сервер продавца HTTP-запрос `GET` следующего вида (см. ниже), при этом:
	• при использовании симметричной криптографии контрольная сумма формируется с помощью ключа, общего для платёжного шлюза и продавца;
	• при использовании асимметричной криптографии контрольная сумма формируется с помощью закрытого ключа, известного только платёжному шлюзу.
	`http://site.ru/path?amount=123456&orderNumber=10747&checksum=DBBE9E54D42072D8CAF32C7F660DEB82086A25C14FD813888E231A99E1220AB3&mdOrder=3ff6962a-7dcc-4283-ab50-a6d7dd3386fe&operation=deposited&status=1`
	Порядок параметров в уведомлении может быть произвольным.
	2. На стороне продавца из строки параметров уведомления удаляется параметр `checksum` и `sign_alias`, а значение этого параметра (контрольная сумма) сохраняется для проверки подлинности уведомления.
	3. Из оставшихся параметров и их значений генерируется строка следующего вида.
	`имя_параметра1;значение_параметра1;имя_параметра2;значение_параметра2;...;имя_параметраN;значение_параметраN;`

При этом пары `имя_параметра;значение_параметра` должны быть отсортированы в прямом алфавитном порядке по имени параметров.

Пример сгенерированной строки параметров представлен ниже.

```
amount;123456;mdOrder;3ff6962a-7dcc-4283-ab50-a6d7dd3386fe;operation;deposited;orderNumber;10747;status;1;
```

4. На стороне продавца высчитывается контрольная сумма, способ вычисления зависит от способа её формирования:
    a. при использовании симметричной криптографии - с помощью алгоритма HMAC-SHA256 и общего с платёжным шлюзом закрытого ключа;
    b. при использовании асимметричной криптографии - с помощью алгоритма хеширования, который зависит от способа создания ключевой пары, и открытого ключа, который связан с закрытым ключом, находящимся на стороне платёжного шлюза.
5. В получившейся строке контрольной суммы все буквы нижнего регистра заменяются на буквы верхнего регистра.
6. Происходит сравнение полученного значения с контрольной суммой, извлечённой ранее из параметра `checksum`.
7. Если контрольные суммы совпадают, сервер отправляет в платёжный шлюз HTTP-код `200 OK`.

Если контрольные суммы совпадают, это уведомление подлинно и было отправлено платёжным шлюзом. В противном случае вероятно, что злоумышленник пытается выдать своё уведомление за уведомление платёжного шлюза.

Если в платёжный шлюз возвращается ответ, отличный от HTTP-кода 200 OK, отправка уведомления считается неуспешной. В этом случае платёжный шлюз повторяет отправку уведомления с определенным интервалом. Повтор отправки уведомлений будет прекращен при достижении следующих условий:

- платёжный шлюз получает HTTP-код 200 OK в ответ на callback-уведомление или
- происходит максимальное количество неуспешных попыток информирования подряд.

По достижении одного из указанных выше условий попытки отправки callback-уведомлений об операции прекращаются

5 Примеры кода

5.1 Пример для Java (симметричная криптография)

```
import org.apache.commons.codec.binary.Hex;

import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.nio.charset.StandardCharsets;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;

public class Example {

    public static String generateHMacSHA256(final String key, final String data)
throws InvalidKeyException, NoSuchAlgorithmException {

        final Mac hMacSHA256 = Mac.getInstance("HmacSHA256");
        byte[] hmacKeyBytes = key.getBytes(StandardCharsets.UTF_8);

        final SecretKeySpec secretKey = new SecretKeySpec(hmacKeyBytes, "HmacSHA256")
;
        hMacSHA256.init(secretKey);

        byte[] dataBytes = data.getBytes(StandardCharsets.UTF_8);
        byte[] res = hMacSHA256.doFinal(dataBytes);

        return new String(Hex.encodeHex(res));
    }

    public static void main(String[] args) throws NoSuchAlgorithmException,
InvalidKeyException {
        String secretToken = "123";
        String message = "amount;1500;mdOrder;ed6f3abf-cea0-427e-
afdf-0ba43ead124f;operation;deposited;orderNumber;89312;status;1;";

        String signature = Expample.generateHMacSHA256(secretToken,
message).toUpperCase();
        System.out.println(signature);
    }
```

5.2 Пример для Java (асимметричная криптография)

```
package ru.bpc.test;


import org.apache.commons.codec.binary.Base64;

import org.apache.commons.codec.binary.Hex;


import java.io.ByteArrayInputStream;

import java.io.InputStream;

import java.security.Signature;

import java.security.cert.CertificateFactory;

import java.security.cert.X509Certificate;

import java.util.Comparator;

import java.util.Map;

import java.util.stream.Collector;

import java.util.stream.Collectors;

import java.util.stream.Stream;


public class App99 {

    public static void main(String[] args) throws Exception {

        String callbackParamsString = "amount=35000099, sign_alias=SHA-256 with RSA,
checksum=163BD9FAE437B5DCDAAC4EB5ECEE5E533DAC7BD2C8947B0719F7A8BD17C101EBDBEACDB295C1
0BF041E903AF3FF1E6101FF7DB9BD024C6272912D86382090D5A7614E174DC034EBBB541435C80869CEED
1F1E1710B71D6EE7F52AE354505A83A1E279FBA02572DC4661C1D75ABF5A7130B70306CAFA69DABC2F620
0A698198F8, mdOrder=12b59da8-f68f-7c8d-12b5-9da8000826ea, operation=deposited,
status=1";
```

```
        Map<String, String> callbackParamsMap =
Stream.of(callbackParamsString.split(","))
                .map(String::trim)
                .map(s -> s.split("="))
                .collect(Collectors.toMap(s -> s[0].trim(), s -> s[1].trim()));


        String checksum = callbackParamsMap.get("checksum");
        callbackParamsMap.remove("checksum");
        callbackParamsMap.remove("sign_alias");


        String signString = callbackParamsMap.entrySet().stream()
                .sorted(Map.Entry.comparingByKey(Comparator.naturalOrder()))
                .collect(Collector.of(
                        StringBuilder::new,
                        (accumulator, element) -> accumulator
                                .append(element.getKey()).append(";")
                                .append(element.getValue()).append(";"),
                        StringBuilder::append,
                        StringBuilder::toString
                ));


        String cert =
"MIICcTCCAdqgAwIBAgIGAWAnZt3aMA0GCSqGSIb3DQEBCwUAMHwxIDAeBgkqhkiG9w0BCQEWEWt6\n" +

"bnRlc3RAeWFuZGV4LnJ1MQswCQYDVQQGEwJSVTESMBAGA1UECBMJVGF0YXJzdGFuMQ4wDAYDVQQH\n" +

"EwVLYXphbjEMMAoGA1UEChMDUkJTMQswCQYDVQQLEwJRQTEMMAoGA1UEAxMDUkJTMB4XDTE3MTIw\n" +

"NTE2MDEyMFoXDTE4MTIwNTE2MDExOVowfDEgMB4GCSqGSIb3DQEJARYRa3pudGVzdEB5YW5kZXgu\n" +
```

```
"cnUxCzAJBgNVBAYTAlJVMRIwEAYDVQQIEwlUYXRhcnN0YW4xDjAMBgNVBAcTBUthemFuMQwwCgYD\n" +

"VQQKEwNSQlMxCzAJBgNVBAsTAlFBMQwwCgYDVQQDEwNSQlMwgZ8wDQYJKoZIhvcNAQEBBQADgY0A\n" +

                "MIGJAoGBAJNgxgtWRFe8zhF6FE1C8s1t/
dnnC8qzNN+uuUOQ3hBx1CHKQTEtZFTiCbNLMNkgWtJ/\n" +

                "CRBBiFXQbyza0/
Ks7FRgSD52qFYUV05zRjLLoEyzG6LAfihJwTEPddNxBNvCxqdBeVdDThG81zC0\n" +

"DiAhMeSwvcPCtejaDDSEYcQBLLhDAgMBAAEwDQYJKoZIhvcNAQELBQADgYEAfRP54xwuGLW/Cg08\n" +

"ar6YqhdFNGq5TgXMBvQGQfRvL7W6oH67PcvzgvzN8XCL56dcpB7S8ek6NGYfPQ4K2zhgxhxpFEDH\n" +

                "PcgU4vswnhhWbGVMoVgmTA0hEkwq86CA5ZXJkJm6f3E/
J6lYoPQaKatKF24706T6iH2htG4Bkjre\n" +

                "gUA=";


        byte[] b = Base64.decodeBase64(cert);

        CertificateFactory certFactory = CertificateFactory.getInstance("X.509");


        InputStream in = new ByteArrayInputStream(b);

        X509Certificate x509Cert =
(X509Certificate)certFactory.generateCertificate(in);


        Signature sig = Signature.getInstance("SHA512withRSA");

        sig.initVerify(x509Cert.getPublicKey());


        sig.update(signString.getBytes());

        boolean verifies =
sig.verify(Hex.decodeHex(checksum.toLowerCase().toCharArray()));

        System.out.println("signature verifies: " + verifies);
```

```
    }

}
```

5.3 Пример для PHP (симметричная криптография)

```
<?php

$data = 'amount;123456;mdOrder;3ff6962a-7dcc-4283-ab50-
a6d7dd3386fe;operation;deposited;orderNumber;10747;status;1;';
$key = 'yourSecretToken';
$hmac = hash_hmac ( 'sha256' , $data , $key);

echo "[$hmac]\n";
?>
```

1. Пропишите строку в переменную `data` .
2. В переменную `key` пропишите закрытый ключ.
3. Функция `hash_hmac` (`'sha256'` , `$data` , `$key`) вычисляет контрольную сумму от переданной строки, с помощью закрытого ключа по алгоритму SHA-256.
4. Сохраните результат работы функции в переменной `hmac` .
5. Выведите результат работы функции функцией `echo` .
6. Сравните это значение с тем, что передано в callback-уведомлении.

5.4 Пример для PHP (асимметричная криптография)

```
<?php
// data from response
$data = 'amount;35000099;mdOrder;12b59da8-
f68f-7c8d-12b5-9da8000826ea;operation;deposited;status;1;';
$checksum =
'9524FD765FB1BABFB1F42E4BC6EF5A4B07BAA3F9C809098ACBB462618A9327539F975FEDB4CF6EC1556F
F88BA74774342AF4F5B51BA63903BE9647C670EBD962467282955BD1D57B16935C956864526810870CD32
967845EBABE1C6565C03F94FF66907CEDB54669A1C74AC1AD6E39B67FA7EF6D305A007A474F03B80FD6C9
65656BEAA74E09BB1189F4B32E622C903DC52843C454B7ACF76D6F76324C27767DE2FF6E7217716C19C53
0CA7551DB58268CC815638C30F3BCA3270E1FD44F63C14974B108E65C20638ECE2F2D752F32742FFC5077
```

```
415102706FA5235D310D4948A780B08D1B75C8983F22F211DFCBF14435F262ADDA6A97BFEB6D332C3D510
10B';

// your public key (e.g. SHA-512 with RSA)
// if you have a CERT, please see openssl_get_publickey()
$publicKey = <<<EOD
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAwtuGKbQ4WmfdV1gjWWys
5jyHKTWXnxX3zVa5/Cx5aKwJpOsjrXnHh6l8bOPQ6Sgj3iSeKJ9plZ3i7rPjkfmw
qUOJ1eLU5NvGkVjOgyi11aUKgEKwS5Iq5HZvXmPLzu+U22EUCTQwjBqnE/Wf0hnI
wYABDgc0fJeJJAHYHMBcJXTuxF8DmDf4DpbLrQ2bpGaCPKcX+04POS4zVLVCHF6N
6gYtM7U2QXYcTMTGsAvmIqSj1vddGwvNGeeUVoPbo6enMBbvZgjN5p6j3ItTziMb
Vba3m/u7bU1dOG2/79UpGAGR10qEFHiOqS6WpO7CuIR2tL9EznXRc7D9JZKwGfoY
/QIDAQAB
-----END PUBLIC KEY-----
EOD;

$binarySignature = hex2bin(strtolower($checksum));
$isVerify = openssl_verify($data, $binarySignature, $publicKey, OPENSSL_ALGO_SHA512);
if ($isVerify == 1) {
    echo "signature ok\n";
} elseif ($isVerify == 0) {
    echo "bad (there's something wrong)\n";
} else {
    echo "error checking signature\n";
}
?>
```