



API интеграции сервиса GAUS

Версия 3.0
15.03.2010

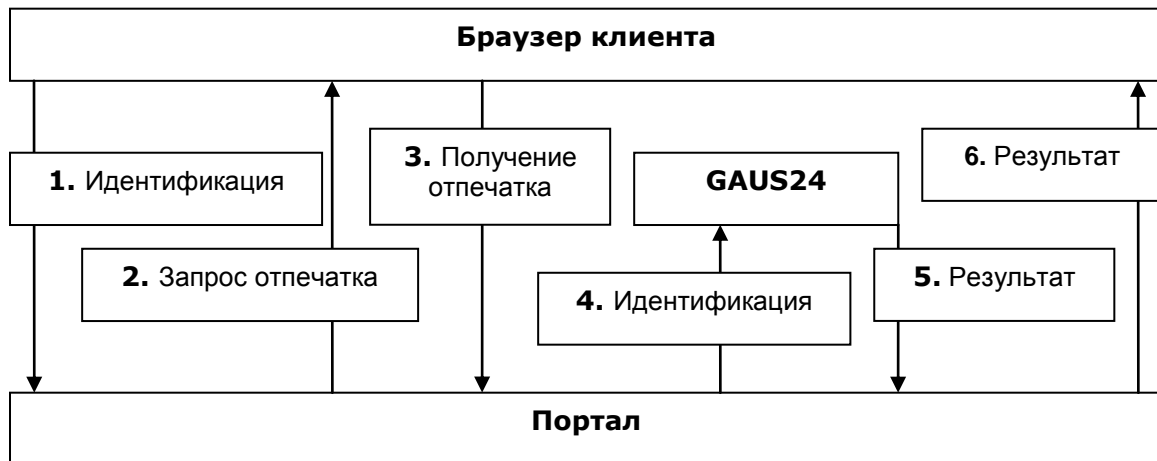
Содержание

1	Общая информация	3
1.1.	<i>Диаграмма потоков данных системы</i>	3
2	Использование GAUS на стороне портала	4
2.1	<i>Регистрация пользователя в системе GAUS (получение сессии GAUS)</i>	4
2.2	<i>Регистрация/Редактирование биометрических данных</i>	5
3	Идентификация пользователя по отпечатку	10
3.1	<i>Сканирование отпечатка на стороне пользователя</i>	10
3.2	<i>Идентификация полученного отпечатка на портале (получение ID по отпечатку)</i>	12
3.3	<i>Вход пользователя в закрытую зону портала</i>	13
4	Примеры использования API на портале	14
4.1	<i>Пример блока управления биометрическими данными</i>	14
4.2	<i>Пример идентификации пользователя по отпечатку</i>	14

1 Общая информация

Целью интеграции системы GAUS на сторонний портал является организация системы биометрической идентификации (идентификации по отпечатку пальца) для простого и безопасного доступа к закрытым разделам сайта или к операциям, требующим подтверждения права пользования.

1.1 Диаграмма потоков данных системы



Пошаговое описание потоков данных:

- 1) Посетитель портала желает идентифицироваться по отпечатку.
- 2) Портал отправляет запрос н.а локальное приложение Gaus.Client посредством браузера пользователя.
- 3) Локальное приложение Gaus.Client производит сканирование отпечатка и отправляет отпечаток на портал по специфическому адресу.
- 4) Портал делает запрос на определение ID пользователя по полученному отпечатку на сервер GAUS.
- 5) GAUS возвращает результат идентификации portalу.
- 6) Результат идентификации обрабатывается порталом и в браузер пользователя возвращается информация относительно дальнейших действий в зависимости от необходимой логики портала.

2 Использование GAUS на стороне портала

Коммуникация портала с системой GAUS осуществляется посредством SOAP-технологии, ниже описаны логические процессы, связанные с GAUS на стороне портала:

1) Регистрация пользователя портала в системе GAUS

Под регистрацией пользователя портала в системе GAUS подразумевается процесс передачи системе уникального идентификатора пользователя в терминах портала и идентификатора портала.

2) Регистрация/редактирование биометрических данных

Данный процесс позволяет зарегистрировать или отредактировать биометрические данные пользователя.

3) Идентификация пользователя на портале

Данный процесс позволяет получить идентификатор пользователя в терминах портала по отпечатку пальца.

Рассмотрим подробно каждый процесс.

2.1 Регистрация пользователя в системе GAUS (получение сессии GAUS)

Под регистрацией пользователя подразумевается процесс передачи системе уникального идентификатора пользователя в терминах портала и e-mail/login пользователя, в результате чего будет получен идентификатор текущей сессии GAUS.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/user?wsdl**

SOAP функция получения сессии выглядит следующим образом:

Result = **GetSession**(PortalUserID, UserMail, PortalID); (для уничтожения сессии используйте функцию **DestroySession**(SessID), в противном случае сессия будет активна в течение 30 мин.)

Где

PortalUserID	– ID пользователя в терминах портала (числовое значение)
UserMail	– Mail пользователя портала
PortalID	– Идентификатор портала (получается интегратором при регистрации портала в системе GAUS)

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code	– Код результата операции
	Данное поле может принимать следующие значения:
140	Ошибка соединения с базой данных
150	Неверный идентификатор портала
230	Пользователь с таким ID не найден

420 Ошибка создания сессии
0 Регистрация успешно завершена, идентификатор сессии в поле **data**

data – Данные, в данном случае идентификатор GAUS сессии или же текстовое описание ошибки

Положительным результатом вызова данной функции является алфавитно-цифровая последовательность, представляющая собой идентификатор GAUS сессии, который может быть использован в дальнейшем для редактирования/регистрации биометрических данных.

PHP пример регистрации пользователя и получения GAUS сессии:

```
// Создаем объект soap клиента
$client = new soapclient(SOAP_CONTENT_SERVER.'/soap/service.php/user?wsdl');

// Регистрация пользователя
$sess = $client->GetUserSession($UserID, $UserMail, PORTAL_ID);

// Проверяем результат
if($sess->result_code) echo "Gaus error: ".$sess->data; else echo "GAUS Session:
    ".$sess->data;
```

2.2 Регистрация/Редактирование биометрических данных

Данный процесс позволяет зарегистрировать или отредактировать биометрические данные пользователя.

Реализация логики регистрации биометрических данных пользователя в целях упрощения процесса интеграции реализована в 2х вариантах, в виде HTML блока с определенными параметрами, который необходимо разместить на странице, отвечающей за биометрические данные пользователя, и в форме API, позволяющего организовать собственную логику и отображение работы с биометрическими данными.

2.2.1 Регистрация/Редактирование биометрических данных в виде HTML блока

Разместите описанный ниже блок на странице личного кабинета пользователя, где должно происходить управление биометрическими данными.

```
<iframe src="<SERVER_URL>/nolang/page/biometry/update_.php/<SESSIONID>">
</iframe>
```

Где

SERVER_URL - это ссылка на сервер системы GAUS,
SESSIONID - это идентификатор GAUS сессии из раздела 2.1

Результатом отображения такого кода будет фрейм с формой управления биометрическими данными для конкретного пользователя. Данный фрейм позволяет также определить специфическую схему идентификации пользователя на портале (подробнее о схемах идентификации читайте [здесь](#)).



Пример фрейма, управляющего биометрическими данными

2.2.2 Управление биометрическими данными посредством API

Управление биометрическими данными подразумевает следующие процессы:

- 1) Регистрация отпечатка пальца пользователя.
- 2) Удаление отпечатка пальца пользователя.

2.2.2.1 Регистрация отпечатков пользователя

Данная функция позволяет зарегистрировать отпечаток пальца для пользователя портала.

Адрес WSDL: <SOAP_CONTENT_SERVER>/soap/service.php/biometry?wsdl

SOAP функция регистрации отпечатка выглядит следующим образом:

Result = **register_fingerprint** (sid, bioData);

Где

- sid** – Идентификатор сессии (из пунктов 2, 3, 4)
- bioData** – Биометрический пакет, полученный от приложения - клиента

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code – Код результата операции

Данное поле может принимать следующие значения:

- 140** Ошибка соединения с базой данных
- 150** Неверный идентификатор портала
- 230** Пользователь с таким ID не найден
- 410** Неверный ключ сессии
- 0** Регистрация успешно завершена

data – Текстовое описание ошибки

2.2.2.2 Получение списка отпечатков

Данная функция позволяет получить список зарегистрированных отпечатков пользователя.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/biometry?wsdl**

SOAP функция получения списка отпечатков выглядит следующим образом:

```
Result = finger_exists(sid);
```

Где

sid – Идентификатор сессии (из пунктов 2, 3)

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code – Код результата операции

Данное поле может принимать следующие значения:

140	Ошибка соединения с базой данных
150	Неверный идентификатор портала
230	Пользователь с таким ID не найден
410	Неверный ключ сессии
0	Успешно выполнено

data – Последовательность 0 и 1, описывающая текущее состояние биометрических данных пользователя или текстовое описание ошибки

2.2.2.3 Удаление отпечатка пальца

Данная функция позволяет удалить отпечаток пальца пользователя портала.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/biometry?wsdl**

SOAP функция удаления отпечатков пользователя выглядит следующим образом:

```
Result = delete_fingerprint(sid, fingerNumber);
```

Где

sid – Идентификатор сессии (из пунктов 2, 3)

fingerNumber – Номер пальца

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code – Код результата операции

Данное поле может принимать следующие значения:

140	Ошибка соединения с базой данных
150	Неверный идентификатор портала
230	Пользователь с таким ID не найден
410	Неверный ключ сессии
0	Успешно удален

data – Текстовое описание ошибки

2.2.2.4 Определение схемы идентификации пользователя

Данная функция предназначена для определения схемы идентификации пользователя на портале.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/biometry?wsdl**

SOAP функция определения схемы идентификации выглядит следующим образом:

Result = **set_scheme** (sid, scheme);

Где

sid – Идентификатор сессии (из пунктов 2, 3)
scheme – Схема идентификации, текстовая строка состоящая из символов «*» и «1»-«10» разделенных “,”

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code – Код результата операции

Данное поле может принимать следующие значения:

140	Ошибка соединения с базой данных
410	Неверный ключ сессии
150	Неверный идентификатор портала
230	Пользователь с таким ID не найден
0	Схема успешно установлена

data – Текстовое описание ошибки

2.2.2.5 Получение текущей схемы пользователя

Данная функция вернет текущую схему идентификации пользователя.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/biometry?wsdl**

SOAP функция получения схемы идентификации пользователя выглядит следующим образом:

Result = **get_scheme** (sid);

Где

sid – Идентификатор сессии (из пунктов 2, 3)

Формат ответа функции:

Ответ функции – объект, содержащий поля **result_code** и **data**

Где

result_code – Код результата операции

Данное поле может принимать следующие значения:

- 140** Ошибка соединения с базой данных
- 410** Неверный ключ сессии
- 150** Неверный идентификатор портала
- 230** Пользователь с таким ID не найден
- 0** Схема успешно получена

data – Текущая схема пользователя, текстовое описание ошибки

3 Идентификация пользователя по отпечатку

Данный процесс позволяет получить идентификатор пользователя в терминах портала по отпечатку пальца и складывается из следующих логических этапов:

- 1) Сканирование отпечатка на стороне пользователя.
- 2) Идентификация отпечатка на портале (получение ID по отпечатку).
- 3) «Залогинивание» пользователя в закрытую зону портала и перенаправление на специфическую страницу.

3.1 Сканирование отпечатка на стороне пользователя

Данный процесс состоит из следующих шагов:

- 1) Получения изображения пальца посредством сканера отпечатка пальцев.
- 2) Преобразование изображения в математический вид.
- 3) Отправка математической формы отпечатка на портал.

Эти 3 процесса выполняются программой Gaus.Client (программа доступна для скачивания по [ссылке](#)), управление данным приложением должно осуществляться логикой портала.

В версии системы > 3.x, приложение Gaus.Client не имеет собственного графического интерфейса, следовательно, задача по визуализации процесса сканирования отпечатков возлагается на конкретный портал, что позволяет отображать GAUS систему в дизайне портала, реализуя необходимую порталу логику.

Для упрощения процесса интеграции, GAUS предоставляет базовое решение (исходный код решения доступен по [ссылке](#)), для работы с приложением Gaus.Client со стороны портала, данная реализация может быть модифицирована на усмотрение портала, ниже несколько слов о реализации.

Для запуска реализации необходимо распаковать архив (архив доступен по [ссылке](#)) в папку на Web сервере, открытую для доступа по HTTP-протоколу.

Архив имеет следующую структуру:

- 1) index.html :
В данном файле показан формат запуска процесса сканирования.
- 2) Папка gaus-client:
 - a. Папка images
Картинки, используемые приложением.
 - b. gausclient.css
Стили приложения.
 - c. gausclient.js
Класс, реализующий логику взаимодействия с приложением Gaus.Client.
 - d. gausclientinterface.js
Класс, реализующий визуализацию приложения на WEB странице. Данный класс содержит русские строковые константы в кодировке UTF-8.

Пример вызова приложения на странице клиента:

....

```
// Подключаем необходимые JS и CSS ресурсы
<script language="javascript" src="../../../gausclient.js" type="text/javascript"></script>
<script language="javascript" src="../../../gausclientinterface.js"
type="text/javascript"></script>
<link rel="stylesheet" href="../../../gausclient.css" type="text/css">
```

```
// Создаем объект отображения
var wgCl = new gausClientInterface();
```

```
// Создаем объект логики
var gCl = new gausClient(wgCl);
```

```
// Запускаем процесс идентификации
gCl.identify("https://<PORTAL_URL>/identify.php");
```

```
// Запускаем процесс регистрации пальца
gCl.register(<номер пальца>"https://<PORTAL_URL>/register.php");
```

....

После выполнения команды **identify/register** будет произведен процесс сканирования отпечатка и отправки математической его формы по адресу **<PORTAL_URL>**, причем сами данные отпечатка будут переданы посредством метода POST, в переменной, под названием **"a"**.

Получив отпечаток пальца, WEB приложение, расположенное по адресу **<PORTAL_URL>** перейдет к следующему шагу – регистрации отпечатка / идентификации пользователя по его отпечатку и произведет логин пользователя или любую другую операцию согласно необходимой логике.

Данные, возвращаемые порталом в приложение Gaus.Client (результат вызова страницы **https://<PORTAL_URL>...** из функций Gaus.Client **identify/register**), несут следующую нагрузку:

- 1) Если страница возвращает целое число – данный ответ воспринимается приложением как код ответа от сервера (например коды **25x**).
- 2) Если страница возвращает нечисловое значение – эти данные будут трактоваться как JavaScript код, который будет доставлен в браузер пользователя и исполнен.
Данный функционал позволяет реализовать произвольную логику реакции на идентификацию или регистрацию клиента.
Пример подобного ответа смотрите ниже.

3.2 Идентификация полученного отпечатка на портале (получение ID по отпечатку)

Под идентификацией пользователя на портале по отпечатку пальца подразумевается процесс передачи системе GAUS уникального идентификатора портала и математической формы отпечатка пальца (полученной из шага 3.1.). Результатом функции является объект, содержащий идентификатор пользователя в терминах портала.

Адрес WSDL: <SOAP_CONTENT_SERVER> /**soap/service.php/biometry?wsdl**

SOAP функция регистрации пользователя выглядит следующим образом:

Result = **identify_fingerprint**(BioData, PortalID);

Где

BioData – Математическая форма отпечатка (из пункта 6.1)
PortalID – Идентификатор портала (получается интегратором при регистрации портала в системе GAUS)

Формат ответа функции:

Ответ функции – объект, содержащий следующие поля

bio_id – Идентификатор пользователя в терминах портала
op_code – Уникальный идентификатор текущей операции идентификации
fng_num – Номер пальца, прошедшего идентификацию
res_code – Код результата операции

Данное поле может принимать следующие значения:

410 Неверный пакет
150 Неверный идентификатор портала
220 Пользователь с таким ID не найден
260 Пользователь идентифицирован, неверная схема
270 Пользователь заблокирован
25x Пользователь идентифицирован, требуется продолжение сканирования
0 Идентификация успешно завершена, идентификатор пользователя в поле **bio_id**

res_text – Текстовое описание ошибки

Следует заметить, что, так как авторизация пользователя на портале может происходить согласно выбранной идентификационной схеме, возможна ситуация, когда необходимо произвести несколько операций сканирования, такая ситуация имеет место, когда значение поля **res_code** находится в интервале от 251 до 259. В этом случае, WEB приложение должно вернуть страницу, содержащую число 25x и ожидать приема следующего биометрического пакета.

3.3 Вход пользователя в закрытую зону портала

Процесс «залогинивания» пользователя строится интегратором системы согласно необходимой логики и структуры конкретного портала на основании полученного идентификатора пользователя из пункта 3.2.

Будучи залогиненным на портале, пользователь получает доступ к редактированию биометрических данных и схемы идентификации, что было описано выше.

В целях снижения нагрузки на систему GAUS, разработчики просят производить запросы, описанные в разделах 2 и 3, только в случае перехода пользователя портала на страницы управления биометрическими данными.

4 Примеры использования API на портале

Ниже расположены примеры, иллюстрирующие реализацию процесса регистрации биометрических данных пользователя и идентификации.

4.1 Пример блока управления биометрическими данными

```
// Создаем объект soap клиента
$client = new soapclient(SOAP_CONTENT_SERVER.'/soap/service.php/user?wsdl');

// Регистрация пользователя
$sess = $client->GetUserSession($UserID, $UserMail, PORTAL_ID);

// Проверяем результат
if($sess->result_code) echo "Gaus error: ".$sess->data; else echo "GAUS Session:
    ".$sess->data;

echo '<iframe src="<SERVER_URL>/nolang/page/biometry/update_.php/'.$sess->data.'>
</iframe>';
....
```

4.2 Пример идентификации пользователя по отпечатку

Для идентификации пользователя необходимо получить отпечаток пальца пользователя. Для этого на странице портала необходимо разместить следующий код:

```
// Подключаем необходимые JS и CSS ресурсы
<script language="javascript" src="../../../gausclient.js" type="text/javascript"></script>
<script language="javascript" src="../../../gausclientinterface.js"
type="text/javascript"></script>
<link rel="stylesheet" href="../../../gausclient.css" type="text/css">

// Создаем объект отображения
var wgCI = new gausClientInterface();

// Создаем объект логики
var gCI = new gausClient(wgCI);

// Запускаем процесс идентификации
gCI.identify("https://<PORTAL_URL>/identify.php");
```

Вызов данного кода отобразит на странице окно процесса сканирования отпечатка пальца, после успешного сканирования отпечаток будет отправлен по ссылке:

https://<PORTAL_URL>/identify.php

На стороне сервера по ссылке

https://<PORTAL_URL>/identify.php
необходимо разместить следующий код:

```
// Получаем биопакет из POST'a
$BioData = @$_POST['a'];

// Создаем SOAP клиент
$client = new soapclient(SOAP_CONTENT_SERVER.'soap/service.php/biometry?wsdl');

// Производим идентификацию клиента
$res = $client->identify_fingerprint($BioData, PORTAL_ID);

// Если ответ = 250, возвращаем 250 и прекращаем исполнение скрипта
if( $res->res_code > 250 && $res->res_code < 260 )
{
    echo $res->res_code;
    return;
}

// Проверяем результат операции
switch($res->res_code)
{

    // Идентификация успешно, перенаправляем пользователя на соотв.
    // страницу
    case 0:
        echo 'location.href="login-successful.html"';
        return;
    // Ошибка идентификации, полный разбор ответа при необходимости.
    default:
        echo 'location.href="login-failed.html"';
        return;
}
}
```