ESET

SECURE AUTHENTICATION

Руководство пользователя API

ESET **SECURE AUTHENTICATION**

Служба поддержки клиентов: www.eset.com/support

Версия от 9. 4. 2015

Содержание

1. Введение

В большинстве веб-приложений пользователи, прежде чем получить доступ к защищенным ресурсам, проходят аутентификацию. Запрашивая дополнительный этап аутентификации при входе в систему, такие приложения получают дополнительный уровень безопасности.

ESET Secure Authentication API — это веб-служба, в основе которой лежит концепция REST и которая позволяет быстро добавлять двухфакторную аутентификацию (2FA) в существующие приложения.

2. Обзор интеграции

Интерфейс API использует две конечные точки:

1. Authentication API для добавления 2FA в существующие приложения.
2. User Management API для управления пользователями 2FA.

API использует способы, которые текст формата POSTing JSON вызывает для соответствующих URL-адресов API. Кроме того, все ответы кодируются в текст формата JSON, который содержит результат метода и все применимые сообщения об ошибках.

Интерфейс API доступен на всех серверах, на которых через защищенный протокол HTTPS на порту 8001 работает установленный компонент Authentication Core. API — это вспомогательный компонент стандартной службы ESA Authentication Service. Домен Active Directory и работающая служба ESA являются обязательными условиями для использования API. Поддерживается только Active Directory. Другие хранилища пользователей нельзя использовать.

3. Настройка

По умолчанию интерфейс API отключен, и его необходимо включить перед использованием. Любой набор учетных данных API можно включить для Authentication API, User Management API или обеих клиентских точек. После включения необходимо создать учетные данные API для авторизации запросов.

1. Запустите ESET Secure Authentication Management Console и перейдите к узлу **Advanced Settings** вашего домена.
2. Разверните раздел **API**.
3. Установите флажок **API is enabled** и, чтобы сохранить изменения, нажмите кнопку **Save**.
4. Запустите ESET Secure Authentication Management Console и перейдите к появившемуся узлу **API Credentials** вашего домена.
5. Чтобы создать новый набор учетных данных, щелкните элемент **Add Credentials**.
6. Дважды щелкните созданные учетные данные, чтобы получить имя пользователя и пароль, которые будут использоваться для аутентификации API. Активируйте для конечных точек нужные учетные данные:
   a. Если учетные данные будут использоваться для Authentication API, установите флажок **Enabled for Auth API**.
   b. Если учетные данные будут использоваться для User Management API, установите флажок **Enabled for User Management API**.
   c. Чтобы сохранить изменения, нажмите кнопку **OK**.
7. Чтобы открыть консоль служб Windows, нажмите клавиши **Windows + R**, введите в поле **Открыть** текст **Services.msc** и нажмите клавишу **Enter**.
8. Правой кнопкой мыши щелкните элемент ESET Secure Authentication **Core** и в контекстном меню выберите команду **Restart**.

Вы можете создать несколько наборов учетных данных API. Для каждого приложения под защитой рекомендуется создать отдельный набор. Также рекомендуется создать отдельный набор для тестирования.

Если интерфейс API включен, все серверы с установленным компонентом Authentication Core после перезапуска будут отвечать на авторизованные запросы API. После создания и удаления учетных записей службу Authentication Core необходимо перезапустить.

4. API аутентификации

Все способы Authentication API доступны по URL-адресам формы *https://127.0.0.1:8001/auth/v1/* и защищены от несанкционированного доступа с помощью HTTP Basic Authentication (для обработки любого запроса требуется действительный набор API Credentials, заранее активированных для Authentication API). Для заголовка *Content-Type* нужно установить значение *application/json* для каждого запроса.

Установщик ESET Secure Authentication автоматически выбирает соответствующий сертификат безопасности SSL, установленный на компьютере. Если сертификат не найден, установщик создает новый самозаверяющий сертификат.

Сведения о замене сертификатов SSL см. в документации, посвященной замене ESA API SSL Certificate.

4.1 Шаг 1: запуск двухфакторной аутентификации

После того как существующее приложение проверит имя пользователя и пароль, необходимо вызвать метод Start 2-FA, чтобы определить, включена ли для пользователя двухфакторная аутентификация. При необходимости в это же время пользователю будет автоматически отправлено SMS OTP.

4.1.1 Запрос

Чтобы начать процесс 2FA, отправьте запрос HTTP POST на следующий URI:

/auth/v1/start2fa

Необходимо опубликовать следующую строку JSON:

```
{
    "username": "USERNAME"
}
```

Поле **username** — это строка с *samAccountName* пользователя, для которого выполняется аутентификация. Следите за тем, чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory.

4.1.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ниже приведен пример стандартного ответа.

```
{
    "expected_otp": ["APP", "SMS"],
    "error": "ERROR_NONE",
    "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться ответ «ERROR_NONE». Описание кодов возможных ошибок см. в разделе Обработка ошибок.

Поле **error_message** содержит доступное описание ошибки, если она возникла.

Поле **expected_otp** — это массив, который обозначает типы OTP, которые пользователь может использовать. Это значение может пригодиться, например, для создания UI, так как от него зависит, должен ли пользователь ожидать SMS или нет. Если массив пуст, это значит, что одноразовый пароль не требуется (т. е. двухфакторная аутентификация не включена) и пользователь может сразу же войти в систему. Массив может содержать следующие типы OTP:

- **APP**: пользователь уже установил приложение ESA на мобильный телефон и должен создать OTP с помощью приложения.
- **SMS**: пользователь не установил приложение, и ему отправлено SMS с OTP.
- **HARD_TOKEN**: пользователю назначен маркер оборудования, и пользователь должен создать OTP с помощью соответствующего устройства.

4.2 Шаг 2: аутентификация

4.2.1 Запрос

Чтобы проверить подлинность пользователя, отправьте запрос HTTP POST на следующий URI:

/auth/v1/authenticate

Необходимо опубликовать следующую строку JSON:

```
{
    "username": "USERNAME",
    "otp": "123456"
}
```

Поле **username** — это строка с именем *samAccountName* пользователя, для которого выполняется аутентификация, а поле **otp** — строка с OTP, который указал пользователь.

4.2.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ниже приведен пример стандартного ответа.

```
{
    "authenticated": true,
    "error": "ERROR_NONE",
    "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

Поле **authenticated** является Boolean, которое указывает, правильно ли введен OTP. Если поле **authenticated** содержит значение **true**, это значит, что OTP пользователя проверен и пользователь может войти в систему.

5. API управления пользователями

Все методы User Management API доступны по URL-адресам формы *https://127.0.0.1:8001/manage/users/v1/* и защищены от несанкционированного доступа с помощью HTTP Basic Authentication (для обработки любого запроса требуется действительный набор API Credentials, заранее активированных для User Management API). Для заголовка *Content-Type* нужно установить значение *application/json* для каждого запроса.

Установщик ESET Secure Authentication автоматически выбирает соответствующий сертификат безопасности SSL, установленный на компьютере. Если сертификат не найден, установщик создает новый самозаверяющий сертификат.

Сведения о замене сертификатов SSL см. в документации, посвященной замене ESA API SSL Certificate.

5.1 Получение профиля пользователя

Этот метод возвращает сведения о 2FA для учетной записи пользователя.

5.1.1 Запрос

Чтобы получить профиль пользователя 2FA, отправьте запрос HTTP GET на следующий URI:

/manage/users/v1/profile/USERNAME

В этом запросе **USERNAME** — это строка с *samAccountName* пользователя, чей профиль нужно получить. Следите за тем, чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory. Имя пользователя должно быть закодировано как URL-адрес.

5.1.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ниже приведен пример стандартного ответа.

```
{
    "username": "USERNAME",
    "mobile_number": "2700000",
    "is_locked": false,
    "last_success": "2014-01-01T00:00:00",
    "last_failure": null,
    "consecutive_failures": 0,
    "credential_type": [ "APP", "SMS" ],
    "error": "ERROR_NONE",
    "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

Поле **username** — это String с *samAccountName* пользователя.

Поле **mobile_number** — это String с мобильным номером пользователя.

Поле **is_locked** — это Boolean, который указывает, заблокирована ли для пользователя 2FA из-за слишком большого количества попыток аутентификации.

Поле **last_success** — это Date, указывающая, когда в последний раз пользователь успешно прошел аутентификацию. Это поле может иметь значение null.

Поле **last_failure** — это Date последней аутентификации пользователя, которая завершилась сбоем. Это поле может иметь значение null.

Поле **consecutive_failures** — это Integer, обозначающее количество последовательных попыток аутентификации, которые выполнял пользователь и которые закончились сбоем.

Поле **credential_type** — это массив, который обозначает типы OTP, включенные для пользователя. Массив может содержать следующие типы OTP:

- **APP** — пользователю открыт доступ к мобильному приложению ESA.
- **SMS** — для пользователя настроена отправка OTPs через SMS.
- HARD_TOKEN — для пользователя настроена отправка OTPs, созданных маркерами оборудования.

5.2 Разблокировка

Этот метод используется для разблокировки 2FA для пользователя. Разблокировать учетную запись, заблокированную службой Active Directory, с его помощью нельзя.

5.2.1 Запрос

Чтобы разблокировать пользователя, отправьте запрос HTTP POST на следующий URI:

/manage/users/v1/unlock

Необходимо опубликовать следующую строку JSON:

```
{
    "username": "USERNAME"
}
```

Поле **username** — это строка с *samAccountName* пользователя, которого нужно разблокировать. Следите за тем, чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory.

5.2.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ответ содержит только код возможной ошибки и сообщение. Ниже приведен пример стандартного ответа.

```
{
    "error": "ERROR_NONE",
    "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

5.3 Отзыв

Этот метод отключает 2FA для пользователя.

5.3.1 Запрос

Чтобы отключить 2FA для пользователя, отправьте запрос HTTP POST на следующий URI:

/manage/users/v1/deprovision

Необходимо опубликовать следующую строку JSON:

```
{
    "username": "USERNAME"
}
```

Поле **username** — это строка с *samAccountName* пользователя, для которого нужно отключить 2FA. Следите за тем, чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory.

5.3.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ответ содержит только код возможной ошибки и сообщение. Ниже приведен пример стандартного ответа.

```
{
    "error": "ERROR_NONE",
    "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

5.4 Подготовка мобильного приложения

Этот метод позволяет пользователям использовать Mobile Application OTPs. Пользователю будет отправлено текстовое сообщение с URL для установки мобильного приложения.

5.4.1 Запрос

Чтобы подготовить пользователя к использованию Mobile Application, отправьте запрос HTTP POST на следующий URI:

/manage/users/v1/provisionmobileapp

Необходимо опубликовать следующую строку JSON:

```
{
    "username": "USERNAME"
}
```

Поле **username** — это строка с *samAccountName* пользователя, которого нужно подготовить. Следите за тем,

чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory.

5.4.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ниже приведен пример стандартного ответа.

```
{
        "installation_url": "http://...",
        "error": "ERROR_NONE",
        "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

Поле **installation_url** — это String, которая содержит URL установки Mobile Application.

5.5 Сообщение о подготовке

Этот метод позволяет пользователям использовать OTPs.

5.5.1 Запрос

Чтобы подготовить пользователя к получению OTPs в текстовых сообщениях, отправьте запрос HTTP POST на следующий URI:

/manage/users/v1/provisiontextmessage

Необходимо опубликовать следующую строку JSON:

```
{
        "username": "USERNAME"
}
```

Поле **username** — это строка с *samAccountName* пользователя, которого нужно подготовить. Следите за тем, чтобы в API было отправлено правильное имя пользователя: *samAccountName* — это имя пользователя для входа в Active Directory.

5.5.2 Ответ

Даже если запрошенное действие не выполнено, все типичные ответы возвращаются с кодом состояния 200 ((OK) HTTP). Ответом является строка JSON. Ответ содержит только код возможной ошибки и сообщение. Ниже приведен пример стандартного ответа.

```
{
        "error": "ERROR_NONE",
        "error_message": ""
}
```

Если ошибки отсутствуют, в поле **error** будет отображаться текст **ERROR_NONE**. Описания кодов возможных ошибок приведены в настоящем руководстве в разделе Обработка ошибок.

Поле **error_message** содержит описание возникшей ошибки.

6. Обработка ошибок

6.1 Ошибки API

Все ошибки API будут возвращены как ответы с кодом состояния HTTP 200 (OK).

Поле **error** в ответе JSON будет содержать код ошибки, который является значением строки литерала. Определены следующие коды ошибок:

- **ERROR_NONE**: ошибок нет;
- **ERROR_USER_NOT_FOUND**: указанное имя пользователя не существует в системе;
- **ERROR_FAULT**: произошла неизвестная ошибка.

Помимо поля **error**, также есть поле **error_message**, в котором указывается понятное описание ошибки. Для определения условий ошибки следует использовать только поле **error**, так как поле **error_message** служит только информационным целям и может быть изменено без предварительного уведомления.

6.2 Ошибки HTTP

Все ошибки HTTP будут возвращаться как ответы без текста или ответы с любым кодом состояния HTTP, кроме 200 (OK).

Могут быть возращены следующие ошибочные коды состояния HTTP:

- **HTTP 500 (Internal Server Error)**: в службе API произошла неизвестная неустранимая ошибка.
- **HTTP 400 (Bad Request)**: неправильный формат заголовка "Authorization" в запросе HTTP.
- **HTTP 401 (Unauthorized)**: в запросе HTTP не указаны учетные данные API.
- **HTTP 403 (Forbidden)**: в запросе HTTP указаны неправильные учетные данные.