DOI 10.37882/2223-2966.2023.12.14

МНОГОФАКТОРНАЯ АУТЕНТИФИКАЦИЯ В ОБЛАЧНЫХ СЕРВИСАХ

MULTI-FACTOR AUTHENTICATION IN CLOUD SERVICES

A. Kozlov

Summary. The article describes the principles of multi-factor authentication and the use of multi-factor authentication in cloud and payment services, discusses practical methods for implementing two-factor authentication.

Multi-factor authentication allows you to reduce information security risks by using several independent factors distributed across various channels.

The article offers a software solution for the implementation of two-factor authentication, in which one-time passwords are used as the second factor.

Keywords: multi-factor authentication, cloud services, PCI DSS, HTTP, HTTP.

Козлов Александр Владимирович
Кандидат технических наук, доцент, РТУ МИРЭА (Российский технологический университет МИРЭА), г. Москва
Email: kozlov.card@gmail.com

Аннотация. В статье описаны принципы многофакторной аутентификации и применение многофакторной аутентификации в облачных и платежных сервисах, рассмотрены практические методы реализации двухфакторной аутентификации.

Многофакторная аутентификация позволяет снижать риски информационной безопасности за счет применения нескольких независимых факторов, распределенных по различным каналам.

В статье предложено программное решение для реализации двухфакторной аутентификации, в которой в качестве второго фактора используются одноразовые пароли.

Ключевые слова: многофакторная аутентификация, облачные сервисы, PCI DSS, TOTP, HOTP.

Введение

Всестороннее технологическое развитие, вывод всё большего количества процессов в «цифровой мир» — одна из отличительных особенностей двадцать первого века.

Однако, избавляя мир от ряда трудностей, неудобств или проблем, цифровизация также привносит и свои, совершенно новые — требующие новых подходов и решений. Облачные сервисы, к примеру, открыли множество возможностей для нарушения безопасности ресурсов пользователей — посредством несанкционированного доступа, хищения или уничтожения их злоумышленниками. Платежные сервисы же ставят под риск один из наиболее критичных типов данных — финансовую информацию клиентов, утрата или разглашение которой всегда приводит к серьезным последствиям для обеих сторон.

Управление доступом к информации — один из наиболее показательных и важных процессов обеспечения безопасности информации. По статистике, порядка семидесяти пяти процентов инцидентов безопасности случаются из-за неправильного управления доступами в скомпрометированной в результате этого системе.

Исследование и анализ проблематики управления доступом позволяет составить минимальный набор требований и рекомендаций к ней, а также разработать «краеугольный камень» всех современных сервисов — веб-приложение многофакторной аутентификации.

Особенности управления доступом в облачной инфраструктуре

Контроль и управление доступом — это часть безопасности, с которой пользователи сталкиваются в первую очередь и больше всего — когда входят в систему на своих компьютерах и мобильных устройствах, делятся файлами или пытаются получить доступ к приложению.

Традиционный подход к управлению доступом основан на ограничении доступа к корпоративной сети, а затем дополнении его другими элементами управления, по мере необходимости. Такая модель ограничивает все ресурсы, подключаемые к корпоративной сети, что делает её слишком ограничительной для полноценного удовлетворения потребностей динамичного предприятия.

Иной подход заключается в том, что организации должны придерживаться «нулевого доверия» в плане контроля и управления доступом, особенно в случаях, когда они практикуют удаленную работу и облачные технологии для цифрового преобразования, и развития своей бизнес-модели, модели взаимодействия с клиентами, модели вовлечения сотрудников и модели расширения прав и полномочий.

Принципы «нулевого доверия» помогают устанавливать и постоянно совершенствовать безопасность информации, сохраняя при этом гибкость, позволяющую идти в ногу с современным миром.

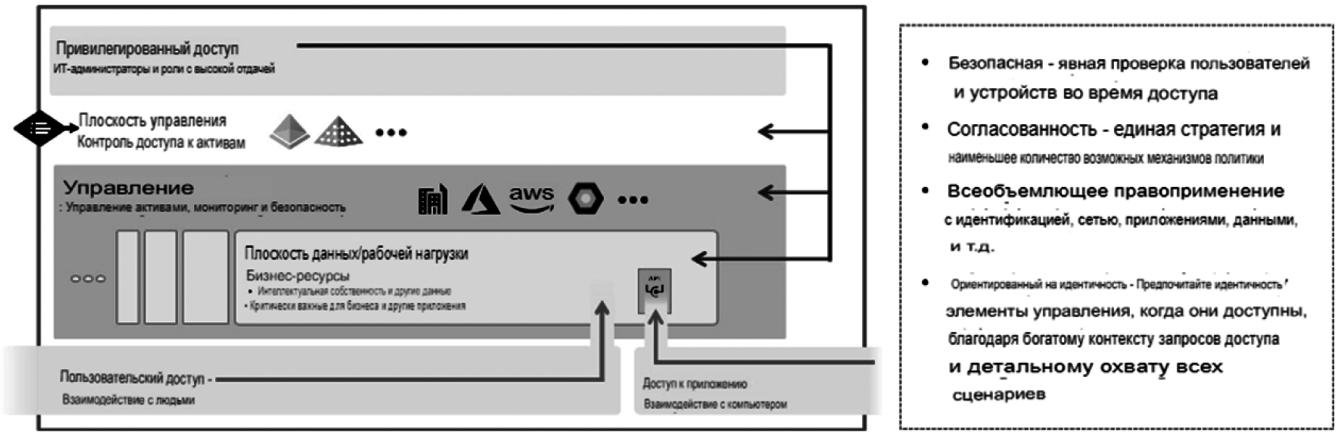

Рис. 1. Схема современной модели доступа

Диаграмма, приведенная выше, демонстрирует все элементы, которые организации должна учитывать при разработке системы управления доступом для нескольких рабочих структур, облаков, различных уровней бизнес-процессов и доступов как людей, так и устройств.

Одним из полезных аспектов организации контроля доступа с «нулевым доверием» является то, что он переходит от статического двухэтапного процесса аутентификации и авторизации к динамическому трехэтапному процессу, также называемому «известный, доверенный, разрешенный»:

— Известный: аутентификация, которая гарантирует, что вы тот, за кого себя выдаете. Этот процесс аналогичен физическому процессу проверки документа, удостоверяющего личность.
— Доверенный: проверка того, что пользователь или устройство достаточно надежны для доступа к ресурсу. Этот процесс аналогичен системе безопасности в аэропорту, которая проверяет всех пассажиров на предмет угроз безопасности, прежде чем разрешить им войти в аэропорт.
— Разрешенный: предоставление определенных прав и привилегий для приложения, службы или данных. Этот процесс аналогичен, к примеру, процессу авиакомпании, которая управляет тем, куда направляются пассажиры, в каком салоне они сидят (первый класс, бизнес-класс или эконом).

На диаграмме (см. рис. 2) демонстрируется механизм политики и точки принудительного применения политики — для возможности реализации подхода «известный, доверенный и разрешенный».

Управление доступом для платежных сервисов

Финансовые услуги и банковская отрасль являются одними из наиболее ценных целей кибератак. Большин-

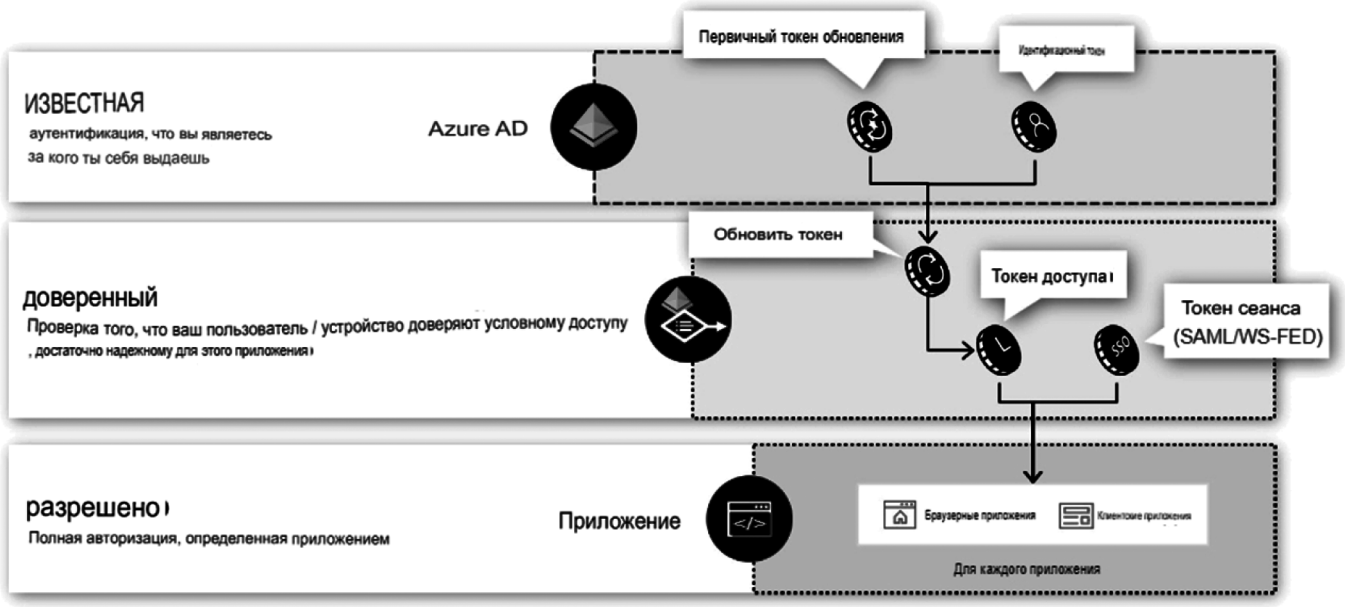

Рис. 2. Ключевые технологии контроля доступа

ство атак на индустрию финансовых услуг использовали фишинг в качестве начальной стартовой площадки. Злоумышленники уже давно признали, что аутентификацию относительно легко взломать, гораздо проще, чем искать и использовать код или недостатки безопасности.

Основным средством обеспечения более надежной аутентификации для организаций, предоставляющих финансовые услуги, является внедрение многофакторной аутентификации [7].

Многофакторная аутентификация — это система безопасности, требующая нескольких учетных данных для аутентификации пользователя. Вместо простого имени пользователя и пароля для MFA требуются дополнительные данные, такие как токен, сгенерированный на пользовательском устройстве, ответ на секретный вопрос, отпечаток пальца или распознавание лица.

PCI DSS [1,2] требует имплементации многофакторной аутентификации для удаленного доступа к среде данных держателей карт.

Процесс аутентификации MFA включает два или более из методов аутентификации, указанных в требовании PCI DSS 8.2. Факторы аутентификации, определенные для MFA, следующие:

— Фактор знания (что-то, что вы знаете). Этот метод включает проверку предоставленной пользователем информации, такой как пароль, PIN-код или секретный ответ на вопрос.
— Фактор владения (то, что у вас есть). Этот метод включает проверку определенного элемента, принадлежащего человеку — токен безопасности, одноразовый пароль, ключ, карта доступа или SIM-карта для телефона. Для мобильной аутентификации элемент владения обычно предоставляется смартфоном в сочетании с программным обеспечением OTP или устройством, содержащим криптографический контент.
— Фактор поведения (что-то, чем вы являетесь). Этот метод включает проверку специфических для человека функций, таких как сканирование сетчатки, сканирование радужной оболочки, сканирование отпечатков пальцев, сканирование вен пальцев, распознавание лиц, распознавание голоса и геометрия вен.

Методы аутентификации, используемые для MFA, должны быть такими, чтобы компрометация одного фактора не позволяла получить доступ к другому. Кроме того, использование любого метода должно быть независимо друг от друга, чтобы не влиять на целостность или конфиденциальность других факторов.

Внеполосная аутентификация относится к процессам аутентификации, в которых методы аутентификации распределены по различным сетям или каналам. Другими словами, факторы, передаваемые по отдельным защищенным каналам, определяются как «внеполосная аутентификация».

В случаях, когда факторы аутентификации передаются по одному каналу, злоумышленник, получивший контроль над устройством, будет иметь возможность обойти все факторы аутентификации.

Термины многофакторная аутентификация и многоступенчатая аутентификация часто путают или используют как взаимозаменяемые. Однако эти два метода аутентификации имеют разные функции, и понимание разницы между ними также очень важно для безопасности доступа.

Многофакторная аутентификация относится к использованию нескольких форм аутентификации. Для аутентификации следует использовать как минимум два различных фактора. В случае, если злоумышленники украдут ваш пароль, они также должны пройти совершенно другой метод аутентификации, чтобы получить доступ.

Существует несколько этапов многоступенчатой аутентификации. Однако шаги используют одну и ту же форму аутентификации. Использование только двух шагов вместо разных факторов при аутентификации называется двухэтапной аутентификацией.

Такие сценарии, как использование двух разных паролей для аутентификации на устройстве или использование двух форм биометрической аутентификации, таких как сканирование сетчатки глаза и снятие отпечатков пальцев в системе, являются примерами многоэтапной аутентификации.

Многофакторная аутентификация более безопасна, чем многоступенчатая аутентификация. Чтобы получить доступ к любому из ваших устройств, злоумышленникам необходимо получить доступ к вашей личной информации, владеть физическим элементом или выдавать себя за ваши биометрические данные.

Аутентификация через один канал снижает эффективность многофакторной аутентификации.

PCI DSS [8,9] рекомендует использовать внеполосную аутентификацию для повышения уровня обеспечения MFA.

Приложения многофакторной аутентификации

Одноразовый пароль на основе времени — распространенный способ реализации двухфакторной аутен-

тификации в приложениях [6]. Он работает, запрашивая у пользователя токен, обычно отправляемый в виде SMS, электронной почты или сгенерированного секретного пропуска на устройство пользователя со сроком действия. Он сравнивает предоставленный токен с фактически сгенерированным токеном, а затем проверяет их подлинность, если токены совпадают.

Приложения-аутентификаторы используют программный метод аутентификации, который реализует 2FA с использованием TOTP [3,5] и одноразового пароля на основе HMAC (HOTP) для аутентификации пользователей приложения. Это часть открытой аутентификации (OATH).

TOTP [4,5] — это алгоритм, основанный на HOTP, который генерирует одноразовый пароль из общего секретного ключа K и текущей временной метки T с использованием хэш-функции H.

Общий секретный ключ K представляет собой строку Base32, сгенерированную случайным образом или производную, известную только клиенту и серверу, отличающуюся и уникальную для каждого токена.

Больший размер временного шага означает большее окно действия для OTP, которое будет принято системой проверки. Больший размер временного шага открывает большее окно для атаки. В целом, большее окно временного шага означает более длительное время ожидания для пользователя, чтобы получить следующий действительный OTP после последней успешной проверки OTP.

Алгоритм требует хэш-функции H для генерации дайджеста и усечения его для получения пароля из D цифр.

Значение, отображаемое на токене, должно быть легко прочитано и введено пользователем: для этого требуется, чтобы значение было разумной длины. Значение должно состоять по крайней мере из 6 цифр. Также желательно, чтобы значение было «только числовым», чтобы его можно было легко вводить на устройствах с ограниченным доступом, таких как телефоны.

Рекомендуемые настройки, направленные на обеспечение наилучшего компромисса между безопасностью и удобством использования, следующие [10]:

— X (временной шаг): 30 секунд
— D (длина пароля): 6 цифр
— H (хэш-функция): SHA-1
— |K| (длина секретного ключа): 160 бит

В результате каждый пароль представляет собой 6-значное число, полученное из 20-байтового дайджеста SHA-1 и действительное в течение 30-секундного временного шага, за который он был сгенерирован.

$$TOTP(K, C) = Truncate(HMAC\text{-}SHA\text{-}1(K, C))$$

Когда сервер проверки получает OTP от клиента, он самостоятельно вычисляет OTP, используя общий секретный ключ K и его текущую временную метку (не временную метку, используемую клиентом), и сравнивает OTP: если они сгенерированы в течение одного и того же временного шага, они совпадают, и проверка завершается успешно.

Практическая реализация МФА

Для реализации и демонстрации многофакторной аутентификации с использованием вышеупомянутых методов используются следующие программные решения:

— Язык программирования Python версии 3.6;
— Программная среда для работы с Python — PyCharm;
— Текстовый редактор для работы с HTML кодом — Блокнот;
— Базовые знания о работе Flask-приложений.

Двухфакторная аутентификация обычно используется в веб-приложениях в качестве дополнительного уровня безопасности при доступе пользователей к серверу. Используя Python реализовано приложение Flask, защищённое с помощью двухфакторной аутентификации.

Для начала устанавливаем веб-фреймворк Flask, Flask-Bootstrap и библиотеку прототипов, которые используются для сборки сервера и реализации двухфакторной аутентификации.

В приведенном ниже коде реализован сервер Flask, который отображает текст «Peter Babansky KTSO-0317!» при открытии индексной страницы.

```
1  # Импорт необходимых библиотек
2  import pyotp
3  from flask import *
4  from flask_bootstrap import Bootstrap
5  # настройка приложения flask
6  app = Flask(__name__)
7  app.config["SECRET_KEY"] = "APP_SECRET_KEY"
8  Bootstrap(app)
```

Рис. 3. Программный код Flask-сервера

```
15 # маршрут страницы логина
16 @app.route("/login/")
17 def login():
18     return render_template("login.html")
```

Рис. 4. Однофакторная аутентификация в Flask

```python
# маршрут формы для логина
@app.route("/login/", methods=["POST"])
def login_form():
    # данные для демонстрации
    creds = {"username": "...", "password": "..."}

    # форма для ввода данных
    username = request.form.get("username")
    password = request.form.get("password")

    # аутентификация введенных данных
    if username == creds["username"] and password == creds["password"]:
        # перенаправление пользователя на страницу МФА, если данные верны
        flash("Подтвердите, что это действительно Вы", "warning")
        return redirect(url_for("login_2fa"))
    else:
        # уведомление пользователя, если данные неверны
        flash("Учетные данные неверны", "danger")
        return redirect(url_for("login"))
```

Рис. 5. Маршрут страницы логина

В представленном ниже коде реализована возможность аутентификации пользователей с использованием имени пользователя и пароля. Создан файл страницы с именем login.html, а также написан маршрут для обработки запросов, отправленных на страницу входа в систему, и проверки их подлинности.

Для генерации OTP с помощью PyOTP, необходимо создать экземпляр класса TOTP библиотеки PyOTP и вызвать метод now.

PyOTP также предоставляет вспомогательную библиотеку для генерации секретных ключей для инициализации классов TOTP и HOTP:

Далее приведен код для предоставления пользователю страницы для настройки TOTP 2FA. Обновлён маршрут входа в систему в app.py файл для перенаправления пользователей на страницу 2FA после успешной аутентификации. Создан маршрут login_2fa, который отвечает за обработку TOTP 2FA, а также файл с именем login _2fa.html.

```python
# маршрут страницы МФА
@app.route("/login/mfa/")
def login_2fa():
    # генерация случайного секретного ключа для аутентификации
    secret = pyotp.random_base32()
    return render_template("login_2fa.html", secret=secret)
```

Рис. 6. настройка 2 FA

Таким образом, в статье были проанализированы основные подходы к многофакторной аутентификации и приведены примеры ее практической реализации.

Перечень сокращений и обозначений

PCI DSS (Payment Card Industry Data Security Standard) — Стандарт безопасности индустрии платежных карт

MFA (Multi-factor authentication) — Многофакторная аутентификация

2FA (Two-factor authentication) — двухфакторная аутентификация

SFA (Single-factor authentication) — однофакторная аутентификация

CDE (Cartholder Data Environment) — среда данных держателей карт

OTP (One-time password) — одноразовый пароль

TOTP (Time-Based One-time Password) — Одноразовый пароль на основе времени

HOTP (HMAC-Based One-time Password) — Одноразовый пароль на основе HMAC

HMAC (Hash-Based message authentication code) — код проверки подлинности сообщений, использующий хеш-функции

PIN-code (Personal Identification Number) — персональный идентификационный номер

ЛИТЕРАТУРА

1. Стандарт PCI DSS 3.2.1 — 2018 [электронный ресурс]. — URL: https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf?agreement=true&time=1652366138174.
2. Стандарт PCI DSS 4.0 — 2022 [электронный ресурс]. — URL: https://www.pcisecuritystandards.org/documents/PCI-DSS-v4_0.pdf?agreement=true&time=1652366165196.
3. D. M'Raihi. HOTP: An HMAC-Based One-Time Password Algorithm, 2005, 35 с [электронный ресурс]. — URL: https://datatracker.ietf.org/doc/html/rfc4226
4. D. M'Raihi. TOTP: Time-Based One-Time Password Algorithm, 2011, 14 с [электронный ресурс]. — URL: https://datatracker.ietf.org/doc/html/rfc6238.
5. Nicola Moretto. Two-factor authentication with TOTP — 2018 [электронный ресурс]. — URL: https://medium.com/@nicola88/two-factor-authentication-with-totp-ccc5f828b6df.
6. Барышева, А.И. Анализ методов аутентификации при использовании банковских карт / А.И. Барышева. — Текст: непосредственный // Молодой ученый. — 2018. — № 18 (204). — С. 45–50. — URL: https://moluch.ru/archive/204/49958/.
7. Вадим Ференец. Банкинг начинается с аутентификации — 2017 [электронный ресурс]. — URL: https://bosfera.ru/bo/banking-nachinaetsya-s-autentifikacii.
8. Surkay Baykara. What's new in PCI DSS 4.0? — 2022 [электронный ресурс]. — URL: https://www.pcidssguide.com/whats-new-in-pci-dss-v4-0/.
9. CorVVin. TOTP (Time-based one-time Password algorithm) — 2020 [электронный ресурс]. — URL: https://habr.com/ru/post/534064/.
10. Very Academy. Flask Tutorial — Build Your First App with Flask and PyCharm — 2021 [электронный ресурс]. — URL: https://www.youtube.com/watch?v=hEJZpA1bhoU&ab_channel=VeryAcademy.

Журнал «Современная наука: актуальные проблемы теории и практики»