packt

Bezpieczeństwo API w praktyce

Strategie ofensywno-defensywne, testy penetracyjne i bezpieczna implementacja interfejsów API

Confidence Staveley
Słowo wstępne: Christopher Romeo

Helion

Tytuł oryginału: API Security for White Hat Hackers: Uncover offensive defense strategies and get up to speed with secure API implementation

Tłumaczenie: Piotr Pilch

ISBN: 978-83-289-2303-4

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
helion.pl/user/opinie/bewapi
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *helion.pl* (księgarnia internetowa, katalog książek)

Printed in Poland.

Spis treści

CZĘŚĆ 2. Ofensywne naruszanie zabezpieczeń interfejsów API

CZĘŚĆ 3. Zaawansowane techniki testowania i naruszania zabezpieczeń interfejsów API

CZĘŚĆ 4. Zabezpieczenia interfejsów API dla specjalistów technicznych od zarządzania

Ewolucja krajobrazu zagrożeń dotyczących interfejsów API i kwestie bezpieczeństwa

Rozdział 2

W czasie gdy w książce kontynuowana jest próba rozwikłania cyfrowej struktury otaczającego nas świata, **interfejsy API** (ang. *Application Programming Interface*) stały się ważną częścią nowoczesnej technologii. Interfejsy API umożliwiły płynną współpracę sieci usług, począwszy od serwisów społecznościowych i usług w chmurze, a skończywszy na aplikacjach mobilnych i urządzeniach **IoT** (ang. *Internet of Things*). Interfejsy API osiągają to dzięki uproszczeniu łączenia różnych komponentów oprogramowania.

Jednak wraz z wygodą, jaką niosą płynne integracja i transformacja cyfrowa, interfejsy API stwarzają też nową przestrzeń dla zagrożeń cyberbezpieczeństwa. Stały się one atrakcyjnym celem dla cyberprzestępców, którzy chcą wykorzystać ich unikalne luki do uzyskania dostępu do informacji poufnych w celu ich kradzieży.

W rozdziale zostanie najpierw prześledzona historia zabezpieczeń interfejsów API. Przyjrzymy się drodze, jaką one pokonały, od chwili ich pierwszego zastosowania do współczesnych czasów, w których są wszechobecne, a jednocześnie w których związany jest z nimi wzrost zagrożeń cyberbezpieczeństwa.

Przyjrzymy się następnie współczesnemu krajobrazowi zagrożeń związanych z interfejsami API, analizując pojawiające się trendy oraz głośne naruszenia zabezpieczeń interfejsów API, co pokazuje, jakie mogą być konsekwencje ignorowania kwestii ich bezpieczeństwa. Omówione zostaną również przyszłe wyzwania związane z bezpieczeństwem interfejsów wynikające z postępu technologicznego.

W tym rozdziale zostaną omówione następujące główne zagadnienia:

- Historyczna perspektywa zagrożeń związanych z bezpieczeństwem interfejsów API.
- Współczesny krajobraz zagrożeń związanych z interfejsami API.
- Pojawiające się trendy w zakresie bezpieczeństwa interfejsów API.
- Wnioski z rzeczywistego włamania do danych interfejsu API.

Pora zatem zacząć!

Historyczna perspektywa zagrożeń związanych z bezpieczeństwem interfejsów API

Interfejsy API odgrywają kluczową rolę w świecie cyfrowym od momentu ich wprowadzenia. Podobnie jak inne technologie interfejsy nieustannie ewoluowały, dostosowując się do zmieniających się potrzeb i środowisk technologicznych. W rezultacie historia bezpieczeństwa interfejsów API to proces adaptacji do zmieniających się zagrożeń i wyzwań wynikających z ich rozwoju.

Początki interfejsów API

Pierwotnie interfejsy API zostały stworzone jako narzędzia umożliwiające komunikację w ramach jednego systemu między różnymi komponentami oprogramowania. Interfejsy oferowały standaryzowany sposób łączenia różnych części oprogramowania, co było widoczne we wczesnych systemach UNIX, w których funkcje API, takie jak realizujące operacje zapisu i odczytu, umożliwiały komunikację między różnymi elementami systemu.

W początkowych latach interfejsy API były wykorzystywane głównie wewnętrznie, a aplikacje działały na scentralizowanych serwerach. Z tego powodu kwestie bezpieczeństwa koncentrowały się przede wszystkim na zabezpieczeniu całego systemu, a nie poszczególnych interfejsów API. W rezultacie zagadnienia związane z ich bezpieczeństwem były proste i miały ograniczony zakres.

Powstanie Internetu i sieciowych interfejsów API

Wraz z pojawieniem się Internetu i usług sieciowych krajobraz technologiczny zaczął się zmieniać. Wprowadzenie przez firmę Microsoft protokołu **SOAP** (ang. *Simple Object Access Protocol*) pod koniec lat 90. umożliwiło korzystanie z interfejsów API za pośrednictwem Internetu, co zapoczątkowało erę *sieciowych interfejsów API* (ang. *web API*). Stworzyło to nowe możliwości udostępniania danych przez aplikacje, pozwalając im na dostęp do funkcjonalności dostarczanych przez inne aplikacje za pośrednictwem sieci.

Pojawienie się sieciowych interfejsów API oznaczało jednak też, że interfejsy są już w większym stopniu narażone na ataki wynikające z szerszego dostępu do Internetu. Początkowo zabezpieczenia tych interfejsów API opierały się na metodach stosowanych w przypadku stron internetowych, takich jak szyfrowanie za pomocą protokołu SSL/TLS oraz uwierzytelnianie użytkowników przy użyciu opcji Basic Authentication protokołu HTTP.

Pojawienie się stylu architektury REST i nowoczesnych interfejsów API

Zaprezentowanie w 2000 roku przez Roya Fieldinga w ramach swojej pracy doktorskiej stylu **REST** (ang. *Representational State Transfer*) zapoczątkowało nową erę dla interfejsów API. REST reprezentuje bezstanową architekturę klient-serwer z jednolitym interfejsem promującym projekty interfejsów API, które są skalowalne, elastyczne i proste. Przejście na styl REST oraz pojawienie się standardu Web 2.0 skoncentrowanego bardziej na treści generowanej przez użytkowników i interaktywności jeszcze bardziej rozszerzyły rolę interfejsów API. Interfejsy API REST szybko zyskały na popularności dzięki łatwości użycia i skalowalności, tworząc fundamenty umożliwiające tworzenie usług internetowych, a później mikrousług.

Rosnące wykorzystanie interfejsów API do integracji usług zewnętrznych w połączeniu ze zwiększającą się popularnością urządzeń mobilnych i IoT doprowadziło do częstszego ujawniania na większą skalę danych poufnych i funkcjonalności za pośrednictwem interfejsów. To z kolei wymusiło rozwój nowych praktyk i protokołów bezpieczeństwa, takich jak OAuth (w przypadku autoryzacji delegowanej).

Era mikrousług, urządzeń IoT i przetwarzania w chmurze

Wraz z rozwojem architektury **mikrousług** interfejsy API stały się standardowym sposobem komunikacji, udostępniając więcej punktów końcowych niż kiedykolwiek wcześniej. Eksplozja urządzeń IoT oraz pojawienie się usług opartych na chmurze uczyniły interfejsy API częścią infrastruktury łączności cyfrowej. Rosnąca złożoność i liczba używanych interfejsów API wymusiły rozwój ich zabezpieczeń w kierunku bardziej szczegółowych modeli bezpieczeństwa opartych na kontekście określanych często mianem modeli **zerowego zaufania** (ang. *zero trust*).

Wprowadzenie w tych czasach **bram interfejsów API** oraz zaadaptowanie nowoczesnych protokołów, takich jak OpenID Connect, zapewniło dodatkowe poziomy bezpieczeństwa. Rozwiązania te obejmują takie kwestie jak zarządzanie ruchem, kontrola dostępu oraz wykrywanie anomalii we wzorcach.

Porównajmy, jak niektóre z powyższych kwestii są rozwiązywane w przypadku tradycyjnego modelu bezpieczeństwa oraz modelu zerowego zaufania (tabela 2.1).

Poniższa tabela wyszczególnia kluczowe różnice między tradycyjnymi modelami bezpieczeństwa i modelem zerowego zaufania, który zyskuje na popularności w obecnym krajobrazie zagrożeń. W ostatnim wierszu skupiono się szczególnie na konsekwencjach z punktu widzenia bezpieczeństwa interfejsów API, co staje się coraz ważniejsze w miarę ich dalszego rozpowszechniania. W modelu zerowego zaufania interfejsy API, podobnie jak wszystkie inne zasoby, nigdy nie są traktowane jako zaufane domyślnie. Są one stale weryfikowane, co zawsze zapewnia wysoki poziom bezpieczeństwa.

Tabela 2.1. Porównanie tradycyjnych modeli bezpieczeństwa i modelu zerowego zaufania

Właściwość	Tradycyjne modele bezpieczeństwa	Model zerowego zaufania
Przyjęcie poziomu zaufania	Domyślny poziom zaufania w obrębie sieci.	Brak domyślnego poziomu zaufania niezależnie od lokalizacji w sieci.
Kontrola dostępu	Często oparta na lokalizacji w sieci (adres IP).	Oparta na tożsamości użytkownika, urządzeniu i kontekście.
Uwierzytelnianie	Głównie na granicy sieci.	Ciągłe uwierzytelnianie w trakcie całej sesji.
Widoczność w sieci	Ograniczona widoczność wewnątrz sieci.	Pełna widoczność w całej sieci.
Architektura zabezpieczeń	Model „zamku i fosy" oparty na granicy sieci.	Mikrosegmentacja (dostęp z minimalnymi uprawnieniami).
Reakcja na zagrożenia	Reaktywne działania po wystąpieniu włamania.	Proaktywne działania (reagowanie na zagrożenia w czasie rzeczywistym).
Konsekwencje z punktu widzenia bezpieczeństwa interfejsów API	Interfejsy API wewnątrz sieci są z założenia traktowane jako zaufane.	Interfejsy API są traktowane jako potencjalne wektory ryzyka i stale weryfikowane.

Współczesny krajobraz zagrożeń związanych z interfejsami API

Interfejsy API stały się centralnym elementem dostarczania dynamicznych, cyfrowych interfejsów obsługi ukierunkowanych na użytkownika we wszystkich sektorach, począwszy od finansów i opieki zdrowotnej, a skończywszy na handlu internetowym. Wraz ze wzrostem stopnia integracji usług, zarówno mikrousług w ramach organizacji, jak i integracji z usługami zewnętrznymi, dramatycznie wzrosła liczba interfejsów API zarządzanych przez organizacje (rysunek 2.1).

Rysunek 2.1. Wykresy z raportu State of API Security firmy Salt Security demonstrujące wzrost liczby interfejsów API w ciągu ostatniej dekady

Zwiększone uzależnienie od interfejsów API rozszerzyło powierzchnię ataku, narażając organizacje na nowe rodzaje zagrożeń. Interfejsy API pełniące rolę strażników dostępu do kluczowych danych i usług stały się atrakcyjnym celem dla cyberprzestępców. Wykorzystywanie słabych punktów w interfejsach API stanowi potencjalne skrócenie drogi do cennych danych, co czyni je jednym z głównych punktów w krajobrazie współczesnych zagrożeń. Atakujący często wykorzystują luki pojawiające się podczas projektowania i wdrażania interfejsów API, co prowadzi do nieautoryzowanego dostępu, wycieku danych oraz zakłóceń w działaniu usług. Niektóre z najczęstszych zagrożeń przedstawiono w tabeli 2.2.

W następnym rozdziale zagłębimy się w dokonany przez organizację OWASP przegląd 10 największych zagrożeń w roku 2023 związanych z bezpieczeństwem interfejsów API, co zapewnia doskonałe ramy do zrozumienia najistotniejszych luk w zabezpieczeniach interfejsów API. Szczegółowo zostanie omówiona każda luka z praktycznymi przykładami, scenariuszami wykorzystania oraz potencjalnymi strategiami zapobiegania i ograniczania ryzyka. Na razie przyjrzyjmy się kluczowym aspektom, na które obrońcy powinni zwrócić uwagę, aby chronić interfejsy API w coraz bardziej rozbudowanym środowisku cyfrowym.

Kluczowe kwestie dotyczące bezpieczeństwa interfejsów API w powiększającym się ekosystemie

Organizacje muszą być świadome opisanych przez OWASP wyzwań związanych z bezpieczeństwem, a także innych repozytoriów bezpieczeństwa. Muszą implementować odpowiednie środki ochrony, aby zabezpieczyć swoje interfejsy API oraz zarządzane dane. W celu utrzymania efektywnego poziomu bezpieczeństwa interfejsów API konieczne są regularne oceny zabezpieczeń, modelowanie zagrożeń oraz przestrzeganie najlepszych praktyk branżowych.

Zagrożenia związane z interfejsami API stają się coraz poważniejsze wraz ze zwiększającym się stopniem ich wykorzystania i popularności. Wynika to z następujących powodów:

- **Rozszerzona powierzchnia ataku**. Wraz z rosnącym poziomem wykorzystania interfejsów API rośnie liczba potencjalnych punktów wejścia dla atakujących. Każdy punkt końcowy interfejsu API wskazuje potencjalne zagrożenie, a im więcej punktów końcowych, tym większe prawdopodobieństwo występowania błędów w zabezpieczeniach. Ta rozszerzająca się powierzchnia ataku zwiększa ogólne ryzyko.
- **Złożoność ekosystemu interfejsów API**. Interfejsy API często są uwzględniane w złożonych ekosystemach obejmujących różne systemy, aplikacje i usługi zewnętrzne. Złożoność ta zwiększa prawdopodobieństwo występowania błędów w zabezpieczeniach i błędów konfiguracji. Każdy punkt integracji w ekosystemie reprezentuje słabe ogniwo w łańcuchu zabezpieczeń, wprowadzając elementy ryzyka i stwarzając możliwości przeprowadzenia ataku.
- **Zależność od interfejsu API**. Interfejsy API są często stosowane w procesie tworzenia nowoczesnego oprogramowania, umożliwiając płynną integrację i wymianę danych. Z powodu powszechnego wykorzystania tych interfejsów wszelkie błędy lub luki w interfejsie API mogą wywołać efekt domina w całym ekosystemie. Pojedyncza luka może wpłynąć na wiele systemów i aplikacji, zwiększając ryzyko.

Tabela 2.2. Przegląd 10 największych zagrożeń w roku 2023 (według organizacji OWASP) związanych z bezpieczeństwem interfejsów API

Zagrożenie związane z bezpieczeństwem interfejsów API	Opis
API1:2023 — naruszenie autoryzacji na poziomie obiektu	Interfejsy API często zapewniają punkty końcowe obsługujące identyfikatory obiektów, co stwarza szerokie spektrum problemów z kontrolą dostępu.
API2:2023 — naruszenie uwierzytelniania	Mechanizmy uwierzytelniania często są nieprawidłowo implementowane, umożliwiając atakującym wykorzystanie tokenów uwierzytelniania lub błędów w implementacji w celu podszywania się pod innych użytkowników.
API3:2023 — naruszenie autoryzacji na poziomie właściwości obiektu	Brak lub niewłaściwa weryfikacja autoryzacji na poziomie właściwości obiektu może prowadzić do ujawnienia danych lub ich modyfikacji.
API4:2023 — nieograniczone wykorzystanie zasobów	Obsługa żądań API wymaga zasobów, takich jak przepustowość sieci, zasoby procesora, pamięć i przestrzeń dyskowa. Udane ataki mogą spowodować odmowę usługi lub zwiększenie kosztów operacyjnych.
API5:2023 — naruszenie autoryzacji na poziomie funkcji	Złożone zasady kontroli dostępu mogą prowadzić do błędów autoryzacji, umożliwiając atakującym dostęp do zasobów innych użytkowników.
API6:2023 — nieograniczony dostęp do poufnych procesów biznesowych	Interfejsy API podatne na to zagrożenie ujawniają przepływ biznesowy, taki jak zakup biletu lub opublikowanie komentarza, bez uwzględnienia tego, jak funkcjonalność może negatywnie wpłynąć na działalność biznesową, gdy jest nadmierne stosowana w sposób zautomatyzowany.
API7:2023 — fałszowanie żądań po stronie serwera	Zagrożenie to może wystąpić, gdy interfejs API pobiera zdalny zasób bez weryfikacji identyfikatora URI dostarczonego przez użytkownika.
API8:2023 — błędna konfiguracja bezpieczeństwa	Zaniedbane lub źle zarządzane konfiguracje mogą narażać system na różne wektory ataków.
API9:2023 — nieprawidłowe zarządzanie magazynem	Interfejsy API odsłaniają więcej punktów końcowych niż tradycyjne aplikacje sieciowe, co wymaga aktualnej dokumentacji i inwentaryzacji.
API10:2023 — niebezpieczne korzystanie z interfejsów API	Zaufanie projektantów do danych z zewnętrznych interfejsów API może skutkować słabszymi środkami bezpieczeństwa, zapewniając atakującym cel.

- **Ujawnienie danych poufnych**. Interfejsy API mają przeważnie do czynienia z danymi poufnymi, takimi jak informacje o użytkownikach, dane finansowe lub dane osobowe. Wzrost użycia interfejsów API zwiększa możliwość ujawnienia danych lub włamania się do nich. Luka w interfejsie API lub jego błędna konfiguracja może umożliwić uzyskanie nieautoryzowanego dostępu do tych danych, co może mieć poważne konsekwencje.
- **Ryzyka związane z usługami zewnętrznymi**. W celu zapewnienia funkcjonalności lub wymiany danych interfejsy API często korzystają z zewnętrznych usług, bibliotek lub środowisk. Dokonanie integracji z zewnętrznymi interfejsami API uzależnia przedsiębiorstwa od zasad bezpieczeństwa dostawców. Jeśli zewnętrzny interfejs API zawiera błędy w zabezpieczeniach lub padnie ofiarą ataku, może to spowodować nowe zagrożenia w ogólnym ekosystemie interfejsów API, narażając potencjalnie dane poufne lub integralność systemu.
- **Zmieniające się środowisko zagrożeń**. Krajobraz technologiczny i obawy związane z bezpieczeństwem stale się zmieniają. W miarę wzrostu popularności i poziomu wykorzystania interfejsy API przyciągają uwagę cyberprzestępców, którzy nieustannie szukają nowych sposobów wykorzystania luk. Dynamiczna natura krajobrazu zagrożeń wymaga od organizacji proaktywnego podejścia do kwestii bezpieczeństwa interfejsów API. Obejmuje to adaptowanie solidnych systemów uwierzytelniania i autoryzacji, technik weryfikowania poprawności danych wejściowych i oczyszczania, szyfrowania danych poufnych oraz regularnych ocen i audytów zabezpieczeń, a także pozostawanie na bieżąco z najlepszymi praktykami bezpieczeństwa i analizowanie zagrożeń.

Aby zminimalizować te rosnące zagrożenia, przedsiębiorstwa muszą priorytetowo traktować kwestię bezpieczeństwa interfejsów API na każdym etapie ich rozwoju i wdrożenia. Uwzględnia to zaadaptowanie solidnych środków bezpieczeństwa, takich jak implementacja silnych mechanizmów uwierzytelniania i autoryzacji, dokładne sprawdzanie poprawności i oczyszczanie danych wejściowych użytkownika, szyfrowanie danych poufnych, przeprowadzanie regularnych ocen i audytów bezpieczeństwa oraz aktualizowanie wiedzy na temat najnowszych praktyk bezpieczeństwa i pojawiających się zagrożeń. Dzięki kompleksowemu i proaktywnemu podejściu do bezpieczeństwa interfejsów API organizacje mogą zminimalizować ryzyka związane z ich rosnącym wykorzystaniem i popularnością.

Nowe trendy w zakresie bezpieczeństwa interfejsów API

Nowe trendy związane z bezpieczeństwem interfejsów API wynikają z postępu technologicznego, ewoluującego krajobrazu zagrożeń oraz rosnącego znaczenia interfejsów w procesie tworzenia nowoczesnego oprogramowania. Poniżej omówiono niektóre z tych trendów.

Architektura zerowego zaufania trust w zabezpieczeniach interfejsów API

Architektura zerowego zaufania opiera się na zasadzie braku zaufania w sieci. Ściśle nadzoruje ona dostęp do wszystkich zasobów, w tym interfejsów API, wzmacniając rygorystyczne środki kontroli dostępu i uwierzytelniania. Koncentrując się na tożsamości i dostępie opartym na kontekście, architektura ta minimalizuje ryzyko nieautoryzowanego dostępu i przemieszczania się w sieci. Implementacja zerowego zaufania w interfejsach API sprowadza się do solidnej autoryzacji i spójnego systemu zaufania. Oto często sugerowane przez ekspertów branżowych praktyki dotyczące użycia architektury zerowego zaufania do zabezpieczenia interfejsów API:

- **Silne uwierzytelnianie i autoryzacja**. Wielu ekspertów zwraca uwagę na konieczność użycia zaawansowanych środków bezpieczeństwa, takie jak **uwierzytelnianie wieloskładnikowe** oraz szczegółowe mechanizmy kontroli dostępu. Celem jest zapewnienie, żeby tylko autoryzowani użytkownicy mogli uzyskać dostęp do interfejsów API.
- **Mikrosegmentacja**. Sieć jest dzielona na mniejsze, odizolowane od siebie części w celu ochrony środowisk interfejsów API. Eksperci są przekonani, że technika ta może ograniczać niekorzystne skutki wszelkich możliwych włamań, czyniąc interfejsy API mniej podatnymi na ataki.
- **Ciągłe monitorowanie i analityka**. Architektura zerowego zaufania kładzie silny nacisk na ciągłą czujność i analizę danych w celu wykrywania nietypowych lub podejrzanych działań. Organizacje mogą wykorzystywać zaawansowaną analitykę i narzędzia oparte na uczeniu maszynowym, aby wcześnie rozpoznawać zagrożenia i natychmiast reagować.
- **Zasada najmniejszych uprawnień**. Jest to zasada kluczowa w architekturze zerowego zaufania. Użytkownikom, aplikacjom i usługom przyznawane są jedynie absolutnie minimalne, niezbędne uprawnienia dostępu do interfejsów API. Eksperci są zgodni co do tego, że takie podejście ogranicza ryzyko nieautoryzowanego dostępu i minimalizuje szkody w przypadku przechwycenia jakichkolwiek danych uwierzytelniających.
- **Bezpieczne bramy interfejsów API**. Wielu profesjonalistów zaleca korzystanie z bezpiecznych bram interfejsów API. Są one kluczową częścią architektury zerowego zaufania, a ponadto wymuszają reguły bezpieczeństwa. Bramy dokonują kontroli na potrzeby uwierzytelniania i autoryzacji, a także stosują dodatkowe środki bezpieczeństwa, takie jak ograniczanie liczby żądań, szyfrowanie i analiza ruchu.
- **Analiza zagrożeń związanych z interfejsami API**. Niezbędna jest aktualna wiedza o pojawiających się zagrożeniach i lukach związanych z interfejsami API. Eksperci zalecają integrację z systemami bezpieczeństwa kanałów analitycznych zawierającymi informacje o zagrożeniach, aby umożliwić proaktywną obronę przed stale zmieniającymi się zagrożeniami.
- **Testowanie zabezpieczeń interfejsów API**. Kompleksowa ocena zabezpieczeń interfejsów API w ramach struktury architektury zerowego zaufania jest

sugerowana przez wielu ekspertów. Może to obejmować sprawdzanie luk, testy penetracyjne oraz bieżące oceny bezpieczeństwa. Pozwalają one wykrywać i eliminować wszelkie słabe punkty lub luki w zabezpieczeniach.

Przyjęcie tych zaleceń ekspertów umożliwia organizacjom utworzenie bardziej bezpiecznego i odpornego środowiska interfejsów API zgodnego z zasadami architektury zerowego zaufania.

Wykorzystanie technologii blockchain do wzmocnienia bezpieczeństwa interfejsów API

Będąc nowym pojęciem w świecie bezpieczeństwa interfejsów API, technologia blockchain przyciąga uwagę dzięki swojemu potencjałowi w zakresie weryfikacji tożsamości i kontroli dostępu. To innowacyjne podejście może znacząco wzmocnić integralność i bezpieczeństwo interakcji interfejsów API. Oto niektóre kluczowe spostrzeżenia i opinie ekspertów dotyczące zastosowania tej technologii w kontekście bezpieczeństwa interfejsów API:

- **Niezmienność i bezpieczeństwo**. Technologia blockchain jest znana ze swojej niezmiennej i bezpiecznej natury. Eksperci podkreślają, że te cechy mogą stanowić dodatkową warstwę ochrony dla transakcji interfejsów API, utrudniając atakującym manipulację danymi.
- **Zaawansowane uwierzytelnianie i kontrola**. Technologia blockchain umożliwia zdecentralizowane zarządzanie tożsamościami. W ramach tego modelu użytkownicy odpowiadają za zarządzanie własnymi tożsamościami, co eliminuje konieczność korzystania z centralnych jednostek autoryzacji i zmniejsza ryzyko wystąpienia pojedynczego punktu awarii.
- **Integralność danych i audyt**. Rozproszony rejestr technologii blockchain gwarantuje integralność danych oraz dostarcza wiarygodnej ścieżki audytu. Rejestrowanie przez organizacje w ramach tej technologii interakcji interfejsów API zwiększa przejrzystość i możliwość ich śledzenia, co wspiera procesy audytu i zgodności z regulacjami.
- **Ochrona przed manipulacją**. Oferowane przez technologię blockchain mechanizmy konsensusu, takie jak *proof-of-work* czy *proof-of-stake*, chronią przed nieautoryzowanymi zmianami danych interfejsów API. Sprawdzanie poprawności przez wiele węzłów w sieci zmniejsza ryzyko związane z nieuprawnionymi modyfikacjami.
- **Zdecentralizowana odporność**. Rozproszona struktura technologii blockchain eliminuje pojedyncze punkty awarii i zwiększa odporność systemu. Takie podejście może uczynić interfejsy API bardziej odpornymi na powszechne zagrożenia takie jak ataki DDoS.
- **Wyzwania i kwestie do rozważenia**. Choć potencjalne korzyści są obiecujące, eksperci podkreślają także wyzwania związane z adaptowaniem technologii blockchain na potrzeby bezpieczeństwa interfejsów API. Problemy obejmują skalowalność, wysokie wymagania obliczeniowe, kwestie regulacyjne, zgodność z istniejącymi systemami oraz potrzebę szerszej implementacji i standaryzacji w całej branży.

Podsumowując: zastosowanie technologii blockchain w zakresie bezpieczeństwa interfejsów API oferuje obiecujące możliwości, ale wiąże się z unikalnymi wyzwaniami. Eksploracja tej technologii w tym kontekście nadal wzbudza zainteresowanie, wskazując na potencjalnie znaczący trend w ewolucji bezpieczeństwa interfejsów API. Organizacje rozważające to podejście muszą zastanowić się nad korzyściami oraz kwestiami praktycznymi i technologicznymi.

Pojawienie się ataków zautomatyzowanych i botów

Wraz z rosnącą zależnością od interfejsów API w przypadku kluczowych funkcji biznesowych atakujący stosują zautomatyzowane metody, aby na dużą skalę wykorzystywać luki w zabezpieczeniach. Takie ataki są często przeprowadzane za pomocą botów, skryptów lub innych narzędzi, które mogą wykonywać zadania znacznie szybciej niż osoby podejmujące atak.

Ataki zautomatyzowane polegają na wykorzystaniu skryptów, botów lub innych narzędzi w celu realizowania szkodliwych działań w sposób zautomatyzowany. Można je ogólnie podzielić na następujące kategorie:

- **„Upychanie" danych uwierzytelniających**. Automatyczne testowanie kombinacji nazw użytkowników i haseł w celu uzyskania nieautoryzowanego dostępu.
- **Zbieranie** (ang. *scraping*). Wyodrębnianie dużych ilości danych z interfejsów API.
- **Ataki DoS** (ang. *Denial of Service*). Zalewanie interfejsów API żądaniami w celu sprawienia, żeby nie były w stanie odpowiedzieć.
- **Wykorzystywanie luk w zabezpieczeniach**. Automatyczne identyfikowanie i wykorzystywanie słabych punktów w zabezpieczeniach interfejsów API.

W szczególności boty stają się coraz bardziej zaawansowane i są w stanie naśladować ludzkie zachowanie, aby można było obejść takie środki bezpieczeństwa jak CAPTCHA.

Dlaczego ten trend się nasila?

Kilka elementów napędza wzrost liczby metod automatycznego wykorzystywania luk w zabezpieczeniach. Oto niektóre z nich:

- **Skala ataku**. Ataki zautomatyzowane mogą jednocześnie mieć na celu tysiące punktów końcowych, co z perspektywy atakujących czyni je wysoce efektywnymi.
- **Niskie koszty**. Narzędzia do automatyzacji są łatwo dostępne często za darmo lub po niskiej cenie, co pozwala nawet mniej doświadczonym atakującym przeprowadzać złożone ataki.
- **Techniki unikania wykrycia**. Nowoczesne boty mogą emulować ludzkie zachowanie, co bardziej utrudnia ich wykrywanie i zapobieganie atakom.
- **Szybki rozwój interfejsów API**. Wraz z rosnącą liczbą interfejsów API odpowiednio rozszerza się powierzchnia ataku, co sprzyja atakom zautomatyzowanym.

Konsekwencje dla bezpieczeństwa interfejsów API

Automatyzacja w cyberatakach spowodowała następujące poważne konsekwencje z punktu widzenia bezpieczeństwa interfejsów API:

- **Zwiększone ryzyko**. W ramach ataków zautomatyzowanych można szybko wykorzystywać luki, zwiększać uprawnienia lub wykradać dane.
- **Wyczerpywanie zasobów**. Duża liczba automatycznych żądań może zużywać znaczące zasoby serwerowe, co prowadzi do zwiększenia kosztów i możliwego pogorszenia jakości usług.
- **Problemy ze zgodnością**. Brak możliwości wykrywania ataków zautomatyzowanych i zapobiegania im może prowadzić do problemów ze zgodnością z regulacjami.

Wzrost liczby ataków zautomatyzowanych i botów stanowi istotne i narastające zagrożenie dla bezpieczeństwa interfejsów API. W miarę rozwoju technologii zwiększa się też poziom zaawansowania i możliwości tych zautomatyzowanych narzędzi. Organizacje muszą rozpoznać ten nasilający się trend i przyjąć podejście proaktywne, aby móc wykrywać ataki oraz zapobiegać im i minimalizować ich skutki. Dzięki zrozumieniu natury ataków zautomatyzowanych i zaimplementowaniu odpowiedniej warstwowej strategii obrony organizacje mogą lepiej zabezpieczyć swoje interfejsy API przed zmieniającym się krajobrazem zagrożeń.

Kryptografia postkwantowa w zabezpieczeniach interfejsów API

Kryptografia odporna na komputery kwantowe (ang. *QRC — Quantum-Resistant Cryptography*), znana również jako kryptografia postkwantowa, reprezentuje przełomowe osiągnięcie w świecie zabezpieczeń interfejsów API. W obliczu rozwoju obliczeń kwantowych tradycyjne metody kryptograficzne są narażone na ryzyko stania się przestarzałymi, ponieważ komputery kwantowe mogą potencjalnie łamać powszechnie stosowane algorytmy szyfrowania. Kryptografia QRC oferuje rozwiązanie tego nadchodzącego zagrożenia. Zagłębmy się w mechanizmy działania, korzyści i trudności towarzyszące temu rozwijającemu się trendowi.

Sposób działania kryptografii

Objaśnijmy kolejno zasady działania kryptografii QRC:

- **Nowe algorytmy**. Kryptografia QRC wykorzystuje matematyczne algorytmy, które, jak się uważa, są bezpieczne wobec możliwości komputerów kwantowych. Algorytmy te są projektowane z myślą o erze postkwantowej, w której tradycyjne metody mogą zawieść.
- **Interoperacyjność**. Integracja kryptografii QRC z istniejącymi infrastrukturami zabezpieczeń wymaga starannego planowania i adaptacji, gdyż nowe metody kryptograficzne muszą płynnie współpracować z obecnymi systemami.
- **Ciągłe badania**. Ponieważ obliczenia kwantowe to nadal szybko rozwijająca się dziedzina, kryptografia QRC również podlega ciągłemu rozwojowi. Naukowcy pracują nad pełnym zrozumieniem wpływu technologii kwantowej na bezpieczeństwo.

Potencjalne korzyści

Kryptografia QRC zapewnia, że dane zaszyfrowane pozostaną bezpieczne nawet wtedy, gdy atakujący uzyska dostęp do komputera kwantowego. Kryptografia stanowi zabezpieczenie na przyszłość, w której obliczenia kwantowe mogłyby podważyć istniejące protokoły bezpieczeństwa.

Ponadto złożoność algorytmów odpornych na komputery kwantowe oferuje wyższy poziom ochrony przed potencjalnymi zagrożeniami, zapewniając solidną ochronę danych i komunikacji interfejsów API.

Kryptografia QRC to nowatorskie podejście do bezpieczeństwa interfejsów API, które bezpośrednio odnosi się do przyszłych wyzwań związanych z obliczeniami kwantowymi. Dzięki proaktywnemu adaptowaniu tych metod organizacje mogą wyprzedzać zmiany technologiczne i utrzymać solidne środki bezpieczeństwa interfejsów API w świecie postkwantowym. Jest to ekscytujący obszar rozwoju, który zapowiada nową erę w dziedzinie bezpieczeństwa, ale wymaga przemyślanego planowania i ekspertyzy.

Bezpieczeństwo architektury bezserwerowej w kontekście bezpieczeństwa interfejsów API

Architektura bezserwerowa, w ramach której wykonywanie kodu jest całkowicie zarządzane przez dostawcę chmury, zyskuje coraz większą popularność w różnych zastosowaniach, w tym w procesie tworzenia interfejsów API. Trend ten wprowadza nowe modele w zakresie skalowalności i efektywności kosztowej, ale wiąże się również z unikalnymi kwestiami dotyczącymi bezpieczeństwa. Poniżej dogłębnie omówiono bezpieczeństwo architektury bezserwerowej oraz jej rolę w ochronie interfejsów API.

Architektura bezserwerowa

Przeanalizujmy kluczowe cechy architektury bezserwerowej. Oto one:

- **Brak zarządzania serwerami**. W architekturze bezserwerowej nie ma potrzeby konfigurowania serwerów ani zarządzania nimi. Kod jest wykonywany na żądanie, a projektanci płacą za faktycznie wykorzystane zasoby.
- **Elastyczna skalowalność**. Skalowanie odbywa się automatycznie w zależności od obciążenia bez potrzeby interwencji człowieka.
- **Unikalne środowisko zabezpieczeń**. Brak tradycyjnej infrastruktury serwerowej prowadzi do specyficznych kwestii bezpieczeństwa, które stanowią zarówno zaletę, jak i wyzwanie.

Kluczowe korzyści związane z bezpieczeństwem

Architektura bezserwerowa jest atrakcyjna dla wielu projektantów i organizacji, gdyż zapewnia mnóstwo zalet dotyczących bezpieczeństwa. Oto niektóre z nich:

- **Zmniejszona powierzchnia ataku**. Ponieważ projektanci nie zarządzają serwerami, mniejsza jest powierzchnia potencjalnych ataków ukierunkowanych na system operacyjny i infrastrukturę bazową.

- **Zautomatyzowane zarządzanie poprawkami**. Dostawcy usług chmurowych zazwyczaj obsługują wszystkie aktualizacje i poprawki zabezpieczeń, zmniejszając ryzyko luk wynikających z nieaktualnego oprogramowania.
- **Izolacja między funkcjami**. Funkcje bezserwerowe często działają w izolowanych środowiskach, co minimalizuje ryzyko ruchu bocznego w systemie.

Nowe wyzwania

Podobnie jak w przypadku każdej technologii, adaptowaniu architektury bezserwerowej towarzyszą pewne wyzwania. Przyjrzyjmy się im bliżej:

- **Bezpieczeństwo na poziomie funkcji**. Tradycyjne elementy kontroli zabezpieczeń oparte na sieci mogą nie mieć zastosowania, co wymaga skoncentrowania się na zabezpieczaniu poszczególnych funkcji i ich uprawnieniach.
- **Monitorowanie i rejestrowanie**. Monitorowanie w czasie rzeczywistym może być bardziej skomplikowane w środowiskach bezserwerowych, co wymaga solidnych rozwiązań z zakresu rejestrowania i analityki.
- **Ryzyko związane z elementami zewnętrznymi**. Zależność od dostawców chmury i usług zewnętrznych może wprowadzać nowe ryzyka, które wymagają uwzględnienia i zminimalizowania.
- **Potencjalne błędne konfiguracje**. Architektury bezserwerowe są wysoce konfigurowalne, a niewłaściwe ustawienia mogą przypadkowo ujawnić informacje poufne.

Architektura bezserwerowa reprezentuje zmianę innowacyjną związaną ze sposobem projektowania i wdrażania interfejsów API. Oferuje ona nowe możliwości z zakresu efektywności i skalowalności. Choć model zapewnia konkretne korzyści z zakresu bezpieczeństwa, takie jak zmniejszona powierzchnia ataku oraz zautomatyzowane stosowanie poprawek, stwarza też specyficzne trudności wymagające specjalistycznego podejścia i strategii. Rozpoznanie i skuteczne rozwiązanie tych kwestii pozwalają organizacjom wykorzystać technologie bezserwerowe do rozwoju swoich produktów API bez kompromisów w zakresie bezpieczeństwa.

Analityka behawioralna i profilowanie zachowań użytkowników w kontekście bezpieczeństwa interfejsów API

W obliczu zmieniającego się charakteru zagrożeń dotyczących interfejsów API tradycyjne mechanizmy zabezpieczeń często nie radzą sobie z wykrywaniem nietypowych zachowań. W tym kontekście integracja analityki behawioralnej i profilowania zachowań użytkowników staje się nowoczesnym trendem w przypadku zabezpieczeń interfejsów API. Podejście to opiera się na analizie wzorców i tendencji zachowań użytkowników w celu identyfikacji podejrzanych działań i bardziej efektywnej ochrony interfejsów API.

- **Analityka behawioralna**. Polega na monitorowaniu i ocenie aktywności użytkowników oraz ich interakcji z systemem. Dzięki analityce system uczy się, jakie zachowania są *normalne*, i potrafi wykrywać odchylenia od tego wzorca.
- **Profilowanie zachowań użytkowników**. Dotyczy to tworzenia indywidualnych profili użytkowników na podstawie ich typowych wzorców interakcji. Profilowanie ułatwia skonfigurowanie protokołów bezpieczeństwa i alertów.

W dalszej kolejności przyjrzyjmy się korzyściom.

Kluczowe korzyści

Analityka behawioralna i profilowanie zachowań użytkowników oferują wiele korzyści. Oto niektóre z nich:

- **Wczesne wykrywanie anomalii**. Przy znajomości szczegółów normalnych zachowań system może szybko identyfikować nietypowe wzorce, wychwytując ewentualnie szkodliwą aktywność na wczesnym etapie ataku.
- **Zmniejszenie liczby fałszywych alarmów**. Dopasowane protokoły bezpieczeństwa minimalizują liczbę fałszywych alarmów, co pozwala skupić się na rzeczywiście podejrzanych działaniach.
- **Zwiększony poziom personalizacji**. Profilowanie pozwala systemom zabezpieczeń dostosować się do konkretnych potrzeb i zagrożeń związanych z różnymi użytkownikami lub rolami.

Analityka behawioralna i profilowanie zachowań użytkowników reprezentują proaktywne i dynamiczne podejście do kwestii bezpieczeństwa interfejsów API. Dzięki zrozumieniu typowych interakcji użytkowników z interfejsami API techniki te pozwalają na wczesne wykrywanie nietypowych wzorców i potencjalnych włamań. Jak w przypadku każdej nowej technologii implementacja wymaga uwzględnienia potencjalnych trudności oraz ostrożnej integracji z istniejącymi środkami bezpieczeństwa. Przy pomyślnym wdrożeniu analityka behawioralna może znacząco poprawić efektywność i responsywność systemów zabezpieczeń interfejsów API.

Wnioski z rzeczywistych naruszeń danych przez interfejsy API

Aby lepiej zrozumieć ryzyka związane z bezpieczeństwem interfejsów API, trzeba wyciągnąć wnioski z faktycznych włamań do danych, które miały miejsce w przeszłości. Poniżej przedstawiono kilka przykładów.

Naruszenie danych firmy Uber (2016 r.)

Włamanie do danych firmy Uber z 2016 roku stanowi alarmujący przykład tego, jak zaniedbania w zakresie bezpieczeństwa interfejsów API mogą prowadzić do nieautoryzowanego dostępu i wycieku poufnych informacji osobowych. W tym przypadku atakujący

uzyskali dostęp do danych osobowych około 57 milionów użytkowników i kierowców firmy Uber. Oto kluczowe aspekty bezpieczeństwa interfejsów API, które zostały uwydatnione przez ten incydent:

- **Niewłaściwe zarządzanie dostępem do interfejsów API**. Włamanie zostało zainicjowane przez atakujących, którzy zdobyli dane uwierzytelniające interfejsów API z prywatnego repozytorium GitHub firmy Uber. Uzyskali oni następnie dostęp do systemów zaplecza firmy Uber i wyodrębnili dane użytkowników. Podkreśla to konieczność zabezpieczenia dostępu do interfejsów API oraz starannego zarządzania danymi uwierzytelniającymi.
- **Luki w integracji z elementami zewnętrznymi**. Incydent nasiliła luka po stronie zewnętrznego dostawcy firmy Uber zapewniającego magazyn danych w chmurze. Uwydatnia to znaczenie solidnych praktyk bezpieczeństwa podczas korzystania z usług zewnętrznych za pośrednictwem interfejsów API. Kluczowe jest przeprowadzanie szczegółowej weryfikacji zewnętrznych dostawców, ciągłe monitorowanie i jasne zrozumienie ich środków zabezpieczeń.
- **Braki w szyfrowaniu**. Dane użytkowników, które zostały ujawnione podczas tego włamania, nie były zaszyfrowane, co ułatwiło uzyskanie nieautoryzowanego dostępu. Przypomina to wyraźnie o istotności stosowania solidnych metod szyfrowania dla wszystkich danych poufnych przesyłanych za pośrednictwem interfejsów API.
- **Opóźnienia w odpowiedzi na incydent i ujawnienie informacji**. Zwlekanie przez firmę Uber z ujawnieniem włamania i wynikająca z tego faktu krytyka podkreśliły znaczenie natychmiastowej i transparentnej odpowiedzi. Aby zminimalizować szkody i zachować zaufanie, kluczowe jest posiadanie dobrze zdefiniowanego planu reagowania na incydenty, który uwzględnia przejrzyste wytyczne dotyczące powiadamiania stron objętych incydentem.
- **Konsekwencje związane z regulacjami**. Prawne reperkusje włamania pokazują, że niewłaściwe zabezpieczenia interfejsów API mogą prowadzić do surowych sankcji prawnych i znacznego uszczerbku na reputacji. Przestrzeganie obowiązujących przepisów dotyczących ochrony danych i prywatności, takich jak RODO lub CCPA, jest nie tylko wymagane prawnie, ale również kluczowe dla utrzymania zaufania konsumentów.

Włamanie do danych firmy Uber stanowi przestrogę, zapewniając wnioski w kilku kluczowych obszarach bezpieczeństwa interfejsów API. Począwszy od zabezpieczania dostępu do interfejsów API i stosowania silnych metod szyfrowania, a skończywszy na zapewnianiu bezpieczeństwa po stronie zewnętrznych dostawców w celu szybkiej odpowiedzi na incydent przedstawiony incydent ilustruje wieloaspektowy charakter bezpieczeństwa interfejsów API i podkreśla potrzebę całościowego podejścia, które uwzględnia każdy potencjalny punkt wejścia i lukę w zabezpieczeniach. W miarę jak interfejsy API stają się centralnym elementem interakcji cyfrowych, wnioski wyciągnięte z tego i podobnych incydentów muszą kształtować bieżące strategie i praktyki bezpieczeństwa, aby chronić dane poufne, które obsługują.

Naruszenie danych firmy Equifax (2017 r.)

Firma Equifax, będąca jedną z liczących się agencji zajmujących się raportowaniem kredytowym, doświadczyła włamania do danych, które ujawniło dane osobowe i finansowe około 147 milionów osób. Atakujący wykorzystali lukę CVE-2017-5638 istniejącą w środowisku Apache Struts używanym przez firmę. Oto niektóre kluczowe wnioski wynikające z tego włamania:

- **Zarządzanie lukami**. Włamanie podkreśliło konieczność szybkiego aktualizowania i łatania środowisk interfejsów API w celu zminimalizowania ryzyka wykorzystania ich do ataku.
- **Bezpieczny projekt interfejsu API**. Podczas projektowania API kwestia bezpieczeństwa musi być traktowana priorytetowo, obejmując silne mechanizmy uwierzytelniania, elementy kontroli autoryzacji oraz sprawdzanie poprawności danych wejściowych. Włamanie w firmie Equifax było wynikiem słabych elementów kontroli uwierzytelniania.
- **Wymuszanie kontroli dostępu**. Powinno się zastosować właściwe elementy kontroli, aby zapewnić, że tylko uprawnione podmioty mają dostęp do danych poufnych. Brak takich skutecznych elementów pozwolił w przypadku firmy Equifax pozyskać ogromne ilości informacji.
- **Wykrywanie i reagowanie na incydenty**. Szybkie wykrywanie i reagowanie na incydenty dotyczące bezpieczeństwa mogą zmniejszyć skutki włamania. Wolna reakcja ze strony firmy Equifax oraz ujawnienie naruszenia z opóźnieniem nasiliły efekty włamania.
- **Świadomość ryzyk związanych z elementami zewnętrznymi**. Niezastosowanie przez firmę Equifax znanej poprawki dla środowiska Apache Struts uwidacznia znaczenie zachowania należytej staranności w przypadku komponentów zewnętrznych i potwierdzania ich bezpieczeństwa.

Włamanie do danych firmy Equifax z 2017 roku stanowi godne uwagi studium rzeczywistego zdarzenia, które uwidacznia krytyczne znaczenie kwestii bezpieczeństwa interfejsów API.

Naruszenie danych firmy MyFitnessPal (2018 r.)

Włamanie do danych firmy MyFitnessPal, które miało miejsce w 2018 roku, uwypukla istotne wnioski dotyczące konieczności stosowania solidnych praktyk z zakresu bezpieczeństwa interfejsów API. Firma MyFitnessPal należąca do firmy Under Armour padła ofiarą ataku, który objął swoim zasięgiem około 150 milionów kont użytkowników, w tym nazwy użytkowników, adresy e-mail oraz hasła w postaci haszowanej. Oto kluczowe elementy bezpieczeństwa interfejsów API, które zostały uwidocznione przez ten incydent:

- **Wykorzystywanie luk w punktach końcowych interfejsów API**. Atakujący wykorzystali słabości w punktach końcowych interfejsów API firmy MyFitnessPal, aby uzyskać nieautoryzowany dostęp do kont użytkowników. Punkty końcowe, które często pełnią rolę kanałów umożliwiających interakcję między aplikacjami, w tym przypadku stanowią środek pozwalający uzyskać

nieautoryzowany dostęp. Podkreśla to potrzebę szczegółowego podejścia do kwestii bezpieczeństwa punktów końcowych interfejsów API, które nierzadko przechowują wartościowe dane użytkowników.

- **Niedoskonałości związane z uwierzytelnianiem i autoryzacją**. Włamanie ujawniło słabe miejsca w elementach kontroli uwierzytelniania aplikacji, co umożliwiło obejście ich przez atakujących. Incydent ten przypomina o kluczowym znaczeniu solidnych mechanizmów uwierzytelniania, takich jak uwierzytelnianie wieloskładnikowe. Zaimplementowanie ich może zapewnić istotną przeszkodę nawet wtedy, gdy atakujący mają dostęp do poprawnych danych uwierzytelniających.
- **Bezpieczne tworzenie kodu i zarządzanie poprawkami**. Incydent w firmie MyFitnessPal podkreśla znaczenie stosowania bezpiecznych praktyk tworzenia kod, a także regularnego aktualizowania środowisk i oprogramowania interfejsów API oraz uwzględniania w nich poprawek. Regularne oceny zabezpieczeń, w tym testy penetracyjne, są kluczowym narzędziem w procesie identyfikowania i eliminowania słabych punktów w implementacjach interfejsów API.
- **Brak naruszenia zabezpieczeń danych finansowych**. Godnym uwagi aspektem tego włamania było to, że nie zostały naruszone żadne dane finansowe i numery ubezpieczenia społecznego. Może to być zasługą zastosowania w przypadku takich informacji poufnych rozdzielenia magazynu danych lub zaawansowanych środków ochrony, co odzwierciedla dobre praktyki w zarządzaniu danymi.

Włamanie do danych firmy MyFitnessPal stanowi przestrogę odnoszącą się do potencjalnych luk obecnych w zabezpieczeniach interfejsów API, a ponadto mnóstwa możliwych sposobów ich wykorzystania. Podobnie jak wiele włamań, incydent w tej firmie dostarcza cennych wskazówek dotyczących obszarów potencjalnych słabości oraz konieczności ciągłego analizowania i doskonalenia praktyk bezpieczeństwa. Począwszy od bezpiecznego projektu punktów końcowych interfejsów API i solidnych mechanizmów uwierzytelniania, a skończywszy na bezpiecznych praktykach tworzenia kodu i regularnej ocenie zabezpieczeń wnioski wyciągnięte na podstawie tego incydentu zapewniają plan działania w zakresie rozszerzania zabezpieczeń interfejsów API. W czasach, gdy dane osobowe w coraz większym stopniu są poddawane digitalizacji, ochrona tych informacji ma nadrzędne znaczenie. Incydent w firmie MyFitnessPal jest zarówno ostrzeżeniem, jak i przewodnikiem pozwalającym to zrealizować.

Skandal związany z firmami Facebook i Cambridge Analytica (2018 r.)

Przedstawiony tutaj incydent dotyczy uzyskania przez firmę Cambridge Analytica nieautoryzowanego dostępu do danych osobowych milionów użytkowników serwisu Facebook.

Incydent z 2018 roku, który dotyczył serwisu Facebook oraz firmy konsultingowej Cambridge Analytica zajmującej się marketingiem politycznym, jest wyraźnym przykładem wyzwań związanych z bezpieczeństwem interfejsów API. W tym przypadku doszło do nielegalnego pozyskania i wykorzystania przez firmę Cambridge Analytica danych osobowych milionów użytkowników serwisu Facebook. Choć główny nacisk w kontekście

tego skandalu położono na kwestie prywatności danych i kwestie etyczne, podkreślono także kluczową rolę bezpieczeństwa interfejsów API oraz potencjalne zagrożenia wynikające z udostępniania podmiotom trzecim danych użytkowników za pośrednictwem interfejsów API.

Podczas tego incydentu firma Cambridge Analytica manipulowała interfejsem API serwisu Facebook, aby zgromadzić ogromne ilości danych użytkowników bez wymaganych uprawnień. Ujawniło to błąd w tym interfejsie, który umożliwiał aplikacjom zewnętrznym pozyskanie danych nie tylko od użytkowników korzystających z tych aplikacji, ale również od ich znajomych w serwisie Facebook, co doprowadziło do znacznego naruszenia prywatności.

Centralnym elementem tej sprawy był interfejs API serwisu Facebook, który zapewniał projektantom dostęp do różnych informacji o użytkownikach, takich jak dane osobowe, połączenia ze znajomymi oraz inna aktywność na platformie. Pozwalało to na tworzenie aplikacji i usług, które komunikowały się z serwisem Facebook i uzyskiwały dostęp do informacji użytkowników za ich zgodą. Niemniej jednak firma Cambridge Analytica wykorzystała aplikację o nazwie `thisisyourdigitallife` do zbierania danych nie tylko od użytkowników, którzy pobrali aplikację, ale także od ich znajomych w serwisie Facebook (często bez ich wyraźnej świadomości lub zgody).

Incydent ten uwypuklił kilka kluczowych kwestii związanych z bezpieczeństwem interfejsów API:

- **Oczywista zgoda użytkownika**. Sytuacja pokazała konieczność transparentnego określania i zabezpieczania jednoznacznej zgody użytkowników przy uzyskiwaniu dostępu do ich danych osobowych za pośrednictwem interfejsów API. Ujawniono, że użytkownicy mogą nie być świadomi, w jakim zakresie ich informacje mogą być dostępne i wykorzystywane przez aplikacje zewnętrzne.
- **Nadzór nad interfejsami API i elementy kontroli**. Incydent wywołał pytania dotyczące kontroli serwisu Facebook nad jego interfejsem API oraz dostępu do informacji przez zewnętrznych projektantów. Podkreślono potrzebę silnego nadzoru nad interfejsami API, w tym regularnych ocen bezpieczeństwa i monitorowania w celu identyfikacji i blokowania nieautoryzowanego dostępu lub niewłaściwego wykorzystania danych.
- **Regulacje dotyczące dostępności danych**. Incydent podkreślił znaczenie wymuszania szczegółowych elementów kontroli dostępu w interfejsach API. W tym przypadku interfejs API serwisu Facebook umożliwiał szeroki dostęp do danych użytkowników, co ułatwiło nieautoryzowane gromadzenie i wykorzystywanie prywatnych informacji. Zastosowanie rygorystycznych elementów kontroli dostępu oraz ograniczenie dostępu do danych za pośrednictwem interfejsów API mogą pomóc w ograniczeniu takich zagrożeń.
- **Zarządzanie ryzykiem związanym z podmiotami trzecimi**. Skandal uwidocznił potrzebę starannego zarządzania przez firmy zagrożeniami związanymi z dostępem do danych przez podmioty trzecie za pośrednictwem interfejsów API. Zwrócono uwagę na znaczenie rygorystycznej weryfikacji zewnętrznych projektantów, przestrzegania przepisów dotyczących prywatności danych oraz tworzenia precyzyjnych umów i ograniczeń odnoszących się do wykorzystania danych.

Skandal związany z firmami Facebook i Cambridge Analytica podkreśla kluczowe znaczenie bezpieczeństwa interfejsów API. Incydent ten przypomina, że aby chronić dane użytkowników, firmy muszą sprawować ścisły nadzór, zapewniać wyraźną zgodę użytkowników oraz starannie zarządzać dostępem podmiotów trzecich. Wzięcie sobie do serca tych wniosków pozwoli firmom lepiej zabezpieczać swoje interfejsy API i ograniczać ryzyko włamań do danych oraz naruszeń prywatności.

Podsumowanie

W tym rozdziale prześledzono historię interfejsów API, wyjaśniając, jak one ewoluowały, by stać się integralną częścią naszej przestrzeni cyfrowej. Omówiono również najnowsze trendy oraz jako przestrogę przedstawiono rzeczywiste przykłady głośnych włamań do interfejsów API, które podkreślają znaczenie bezpieczeństwa tych interfejsów.

W kolejnym rozdziale zostaną omówione najpoważniejsze zagrożenia dla bezpieczeństwa interfejsów API oraz sposoby ich minimalizowania.

Dodatkowe źródła informacji

Aby dowiedzieć się więcej o zagadnieniach zaprezentowanych w tym rozdziale, zajrzyj na strony, do których kierują następujące odsyłacze:

- D. Ritchie, K. Thompson, *The UNIX Time-Sharing System*, „The Bell System Technical Journal" 1974, 57(6), s. 1905 – 1930.
- D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, D. Winer, *Simple Object Access Protocol (SOAP) 1.1. W3C Note, 8*, 2000.
- R. Fielding, *Architectural Styles and the Design of Network-based Software Architectures*, rozprawa doktorska, University of California, Irvine, 2000.
- J. Lewis, M. Fowler, *Mikrousługi*, *https://martinfowler.com/articles/microservices.html*, 2014.
- J. Kindervag, *Build Security Into Your Network's DNA: The Zero Trust Network Architecture. Forrester Research Inc*, *https://www.virtualstarmedia.com/downloads/Forrester_zero_trust_DNA.pdf*, 2010.
- Gartner, *Top Strategic Predictions for 2022 and Beyond: Navigating the New Normal*, październik 2021.
- Forrester, *The State of API Security*, listopad 2020.
- Akamai Technologies, *2020 State of the Internet / Security: Web Attacks and Gaming Abuse*, 2021.
- IBM Security, *The State of API Security*, kwiecień 2020.
- Microsoft Azure Blog, *Protecting Your APIs with Azure API Management*, Microsoft, styczeń 2022.

Chroń to, co najcenniejsze, i nie daj się zhakować!

Interfejsy API są siłą napędową innowacji w dziedzinie oprogramowania. Umożliwiają płynną komunikację i wymianę danych między różnymi aplikacjami, usługami i systemami. Wzajemna łączność sprawia też jednak, że interfejsy API stają się atrakcyjnym celem dla napastników usiłujących wykorzystać ich podatności i uzyskać dostęp do chronionych danych.

Ten kompleksowy podręcznik docenią specjaliści do spraw bezpieczeństwa i projektanci aplikacji. Znajdziesz w nim szereg przydatnych informacji na temat testowania API, identyfikowania podatności i ich eliminowania. W książce znalazło się mnóstwo praktycznych przykładów, dzięki którym dowiesz się, jak unikać kontroli uwierzytelniania i autoryzacji, a także jak identyfikować podatności w interfejsach API przy użyciu różnych narzędzi. Nauczysz się też tworzenia rozbudowanych raportów dotyczących wykrytych podatności, a ponadto rekomendowania i stosowania skutecznych strategii ich minimalizowania. Poznasz również strategie zarządzania bezpieczeństwem interfejsów API i dowiesz się, jak je chronić przed najnowszymi zagrożeniami.

W książce:

- najlepsze praktyki i standardy bezpieczeństwa API
- testy penetracyjne i ocena podatności API
- modelowanie zagrożeń i ocena ryzyka w kontekście bezpieczeństwa API
- techniki unikania wykrycia
- integracja zabezpieczeń API z przepływem pracy w ramach metodyki DevOps
- nadzór interfejsów API i zarządzanie ryzykiem

Confidence Staveley specjalizuje się w bezpieczeństwie aplikacji i strategiach związanych z cyberbezpieczeństwem. Posiada wyjątkową umiejętność przystępnego tłumaczenia skomplikowanych zagadnień ze świata cyberbezpieczeństwa. Uzyskała wiele branżowych certyfikatów, takich jak CISSP, CSSLP i CCISO. Jest też założycielką CyberSafe Foundation.

Helion
helion.pl
HELION S.A.
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl
KOD KORZYŚCI
Sięgnij po więcej!
ISBN 978-83-289-2303-4
9 788328 923034
Cena: 99,00 zł
packt