UNIVERSITET AXBOROT TIZIMLARIDA API XIZMATLARINI BOSHQARISH VA XAVFSIZLIGINI TA’MINLASH MODELI

Istamov Bexzod Bafoqulovich

Buxoro davlat universiteti, “Kompyuter tizimlari va ularning dasturiy ta’minoti” ta’lim yo‘nalishi magistranti

E-mail: istamovbexzod@gmail.com

Annotatsiya: Maqolada universitet raqamli infratuzilmasidagi API xizmatlarini markazlashtirilgan boshqarish va xavfsizligini ta'minlash masalasi tadqiq etiladi. Universitet axborot tizimlari (talabalar, o'quv jarayoni, kadrlar, moliya, kutubxona, ilmiy faoliyat) o'rtasidagi ma'lumot almashinuvi REST API orqali amalga oshiriladi, ammo endpointlar ko‘payishi, autentifikatsiya mexanizmlarining xilma-xilligi, kirish huquqlarining noto‘g‘ri taqsimlanishi, monitoringning yetishmasligi va eskirgan versiyalar xavfsizlik muammolarini keltirib chiqaradi. Tadqiqotda API Gateway markazida shakllangan model taklif etiladi, unda autentifikatsiya, avtorizatsiya, rate limiting, endpoint inventarizatsiyasi, jurnal yuritish va monitoring yagona boshqaruv qatlamida birlashtiriladi. Model UML komponentlar, ketma-ketlik va foydalanish holatlari diagrammalari orqali asoslanadi, OWASP API Security Top 10 bilan moslik tahlili, joriy etish bosqichlari va baholash mezonlari keltiriladi. Taklif etilayotgan yondashuv mavjud modullarni qayta qurmasdan, ularning API xizmatlari ustidan markazlashtirilgan nazoratni ta'minlaydi.

Kalit so‘zlar: API, API Gateway, API xavfsizligi, autentifikatsiya, avtorizatsiya, JWT, REST API, rate limiting, monitoring, API inventarizatsiyasi, oliy ta’lim axborot tizimi, UML modellashtirish, OWASP.

Kirish: Universitetlarda raqamli texnologiyalarning joriy etilishi natijasida universitet faoliyatini boshqaruvchi ko'plab axborot tizimlari (talabalar reestri, o'quv jarayoni, kadrlar, moliya-buxgalteriya, kutubxona, ilmiy faoliyat, masofaviy ta'lim) shakllanmoqda. Bu tizimlar turli texnologik platformalarda, turli dasturlash tillarida va ko‘pincha turli jamoalar tomonidan mustaqil ishlab chiqilishi mumkin. Shu sababli ular

o'rtasida axborot almashinuvini ta'minlashda API xizmatlari muhim texnologik vosita sifatida namoyon bo'ladi va zamonaviy universitet raqamli ekotizimining “asosiy integratsiya mexanizmi” vazifasini bajaradi.

Avvalgi tadqiqotlarda universitet dasturiy modullarini yagona axborot makoniga birlashtirish uchun API Gateway, integratsion shina va markazlashtirilgan ma'lumotlar omboridan iborat ko‘p qatlamli arxitektura taklif qilingan, unda API Gateway autentifikatsiya, avtorizatsiya va so'rovlarni marshrutlash funksiyalarini bajarishi ko'rsatilgan [1]. Bunday yechimlar modullararo integratsiyani texnik jihatdan ta’minlasada, API xizmatlarining o‘zini mustaqil boshqariladigan raqamli aktiv sifatida tizimli nazorat qilish masalasi ko'pincha ikkinchi darajali vazifa sifatida qoladi.

Biroq API Gateway mavjudligining o'zi API xizmatlarini to'liq boshqarish va xavfsizligini kafolatlamaydi. Endpointlar sonining ortishi bilan ularning qaysi xizmatga tegishliligi, qanday foydalanuvchilar tomonidan chaqirilishi, qaysi versiyasi faol ekanligi va qanday huquqlar asosida ishlashi ustidan nazoratni tashkil etish zarurati paydo bo'ladi. Amaliyotda bu muammo ko‘pincha nazoratsiz, hujjatlashtirilmagan yoki foydalanishdan chiqarilgan, ammo hali ham faol endpoint shaklida namoyon bo'ladi va bunday holatlar xavfsizlik teshiklarining eng ko'p uchraydigan manbalaridan biri hisoblanadi.

OWASP tomonidan e’lon qilingan 2023-yilgi tavsiyalarda obyekt darajasidagi avtorizatsiyaning buzilishi, autentifikatsiya muammolari, resurslardan cheklanmagan foydalanish, funksiya darajasidagi avtorizatsiya, noto‘g‘ri konfiguratsiya, API inventarizatsiyasining yetarli emasligi va uchinchi tomon API xizmatlaridan xavfsiz foydalanmaslik kabi xavflar alohida ko‘rsatilgan [4]. Bu xavflarning aksariyati texnologik emas, balki boshqaruv xarakteriga ega — ya’ni ular ko'proq API hayotiy siklini kim, qanday tartibda va qaysi siyosatlar asosida boshqarayotganligiga bog'liq.

Shu nuqtai nazardan, mazkur tadqiqotda universitet API xizmatlarini faqat integratsiya vositasi sifatida emas, balki alohida boshqariladigan va nazorat qilinadigan raqamli resurs sifatida qarash taklif etiladi. Maqolaning qolgan qismida avval mavzu bo'yicha mavjud yondashuvlar tahlil qilinadi, so‘ngra taklif etilayotgan model UML

diagrammalari orqali tasvirlanadi, xavfsizlik mexanizmlari batafsil yoritiladi va modelni amaliyotga joriy etish bosqichlari ko'rib chiqiladi.

Mavzu bo‘yicha adabiyotlar tahlili

Universitet axborot tizimlarini integratsiyalash mavzusi bir necha yo‘nalishda o‘rganilgan. Birinchi yo‘nalishda modullararo integratsiyaning umumiy arxitekturaviy modellari taklif etiladi, unda API Gateway, integratsion shina va markazlashtirilgan ma’lumotlar ombori asosiy komponentlar sifatida ko‘rsatiladi [1]. Ikkinchi yo‘nalishda alohida modullar (masalan, kutubxona tizimi) integratsiyasining funksional yechimlari o‘rganiladi [2]. Uchinchi yo‘nalish esa ma’lumotlar sinxronizatsiyasiga asoslangan integratsiya modellariga bag‘ishlangan bo‘lib, unda tizimlar orasidagi ma’lumotlar izchilligini ta’minlash masalalari ko‘rib chiqiladi [3]. Ushbu ishlarning umumiy xususiyati shundaki, ularda API texnologiyasi asosan integratsiya vositasi sifatida qaraladi: e’tibor asosan tizimlar orasidagi ma’lumot almashinuvini texnik jihatdan qanday tashkil etishga qaratiladi. API xizmatlarining o‘zini alohida boshqariladigan, hayotiy sikli, versiyasi va xavfsizlik holati kuzatib boriladigan resurs sifatida ko‘rib chiqish masalasi esa yetarlicha chuqur yoritilmagan.

Xalqaro amaliyotda API xavfsizligi masalasi OWASP API Security Top 10 hujjatida tizimlashtirilgan holda taqdim etiladi. Ushbu hujjatda obyekt darajasidagi avtorizatsiyaning buzilishi eng yuqori xavf sifatida ko‘rsatiladi, undan so‘ng noto‘g‘ri autentifikatsiya, obyekt xususiyatlari darajasidagi avtorizatsiya buzilishi, resurslardan cheklanmagan foydalanish, funksiya darajasidagi avtorizatsiya buzilishi, biznes oqimlariga nazoratsiz kirish, xavfsizlik konfiguratsiyasining noto‘g‘ri tashkil etilishi, API inventarizatsiyasining yetarli emasligi va API resurslaridan xavfsiz bo‘lmagan foydalanish kabi kategoriyalar keladi [4, 5]. Shu bilan birga, bearer tokenlardan foydalanish bo‘yicha RFC 6750 standarti tokenlarni uzatish, saqlash va tekshirish jarayonlariga nisbatan aniq texnik talablarni belgilaydi, xususan TLS dan foydalanish majburiyligini ta’kidlaydi [6].

Sanoat amaliyotida keng tarqalgan API Gateway yechimlari (masalan, Kong, Apigee, Amazon API Gateway kabi platformalar) autentifikatsiya, rate limiting,

marshrutlash va monitoring funksiyalarini plaginlar yoki modullar shaklida taqdim etadi [13]. Bunday platformalarning arxitekturaviy tamoyillari, xususan so‘rovni bir nuqtadan o‘tkazish (single entry point) va siyosatlarni markazlashtirish g‘oyasi — bu, o‘z navbatida, REST arxitekturaviy uslubining bir nechta muhim tamoyillariga asoslanadi [8] — mazkur tadqiqotda taklif etilayotgan modelning konseptual asosini shakllantirishda inobatga olindi. Shuningdek, mikroservis arxitekturalarining xavfsizligi bo‘yicha NIST tomonidan ishlab chiqilgan tavsiyalarda xizmatlararo autentifikatsiya, tarmoq segmentatsiyasi va markazlashtirilgan siyosat boshqaruvi muhim chora sifatida qayd etiladi [10].

Xulosa qilib aytganda, universitet miqyosidagi tadqiqotlarda integratsiya arxitekturasi yetarlicha yoritilgan bo‘lsa-da, shu arxitektura ustida ishlaydigan API xizmatlarini OWASP tavsiyalariga mos, UML vositalari bilan rasmiylashtirilgan holda tizimli tahlil qiluvchi model amaliyotda kam uchraydi — aynan shu bo‘shliqni to‘ldirish mazkur tadqiqotning motivatsiyasini tashkil etadi.

Tadqiqot maqsadi, vazifalari, obyekti va metodlari

Tadqiqotning maqsadi — universitet axborot tizimlarida API xizmatlarini markazlashtirilgan boshqarish va xavfsizligini ta’minlashga mo‘ljallangan funksional modelni ishlab chiqishdan iborat. Bu maqsadga erishish uchun quyidagi vazifalar belgilandi: xavfsizlik muammolarini aniqlash; markazlashtirilgan boshqarish mexanizmini ishlab chiqish; autentifikatsiya va avtorizatsiyani Gateway darajasida tashkil etish; rollarga asoslangan kirish nazoratini o‘rnatish; trafikni cheklash; endpoint va versiyalarning yagona inventarini shakllantirish; chaqiruvlarni jurnalga yozish va monitoring qilish; modelni UML orqali rasmiylashtirish; OWASP Top 10 bilan moslikni baholash; joriy etish bosqichlari va mezonlarini belgilash.

Tadqiqot obyekti sifatida oliy ta’lim muassasalarining API orqali o‘zaro ma’lumot almashuvchi axborot tizimlari olindi. Tadqiqot predmeti esa API xizmatlarini boshqarish, autentifikatsiya va avtorizatsiya, trafikni nazorat qilish, monitoring, jurnal yuritish va xavfsizlikni ta’minlash mexanizmlaridan iborat.

Tadqiqot davomida tizimli tahlil, funksional modellashtirish, arxitektura modellashtirish, UML asosidagi vizual modellashtirish (komponentlar, ketma-ketlik va foydalanish holatlari diagrammalari) [12], API xavfsizlik xavflarini tahlil qilish va komponentlararo ma’lumot oqimini modellashtirish usullaridan foydalanildi. Universitet axborot tizimlari uchun mavjud uch qavatli va ko‘p qatlamli arxitekturalarda REST API hamda JWT asosidagi autentifikatsiya mexanizmlari qo‘llanilgani mavjud tadqiqotlarda ko‘rsatilgan [1]. Xususan, ReactJS + Redux frontend, Laravel backend va Django REST Framework asosidagi autentifikatsiya xizmatidan tashkil topgan arxitekturada JWT access va refresh tokenlaridan foydalanilgan [1, 9]. Mazkur tajriba yangi modelda API xizmatlarini boshqarishning xavfsizlik qatlamini alohida ajratish uchun metodik asos sifatida qabul qilindi.

Universitet API xizmatlarini boshqarish muammolari

1. API endpointlarining nazoratsiz ko‘payishi. Turli modullar mustaqil jamoalar tomonidan, mustaqil rivojlanish sur’atlarida ishlab chiqilganda bir xil ma’lumotga xizmat qiluvchi bir nechta endpointlar paydo bo‘lishi mumkin. Natijada API xizmatlarining yagona ro‘yxatini yuritish murakkablashadi, ba’zi endpointlar hujjatlashtirilmagan holda ishlab turadi (shadow API), bu esa xavfsizlik nazoratidan chetda qolgan kirish nuqtalarining paydo bo‘lishiga sabab bo‘ladi.

2. Autentifikatsiya mexanizmlarining bir xilligi ta’minlanmasligi. Turli modullarda foydalanuvchini tekshirish mexanizmlari turlicha tashkil etilsa (masalan, bir modul sessiya asosida, ikkinchisi JWT asosida, uchinchisi API kalit asosida ishlasa), xavfsizlik siyosatini markazlashtirish qiyinlashadi. Bunday holatda har bir modul o‘z autentifikatsiya mantig‘ini alohida qo‘llab-quvvatlashi kerak bo‘ladi, bu esa xatolik ehtimolini va texnik xizmat ko‘rsatish xarajatlarini oshiradi.

3. Avtorizatsiya nazoratining noto‘g‘ri tashkil etilishi. Foydalanuvchi tizimga kirgan bo‘lishi uning barcha API xizmatlaridan foydalanish huquqiga ega ekanligini anglatmaydi — talaba, o‘qituvchi, administrator va boshqa xodimlarning API darajasidagi huquqlari farqlanishi kerak. Amaliyotda avtorizatsiya nazorati ko‘pincha faqat interfeys (frontend) darajasida amalga oshiriladi, API darajasida esa

tekshirilmaydi — bu OWASP tomonidan “obyekt darajasidagi avtorizatsiyaning buzilishi” nomi bilan eng xavfli tahdid sifatida qayd etiladigan holatni yuzaga keltiradi [4].

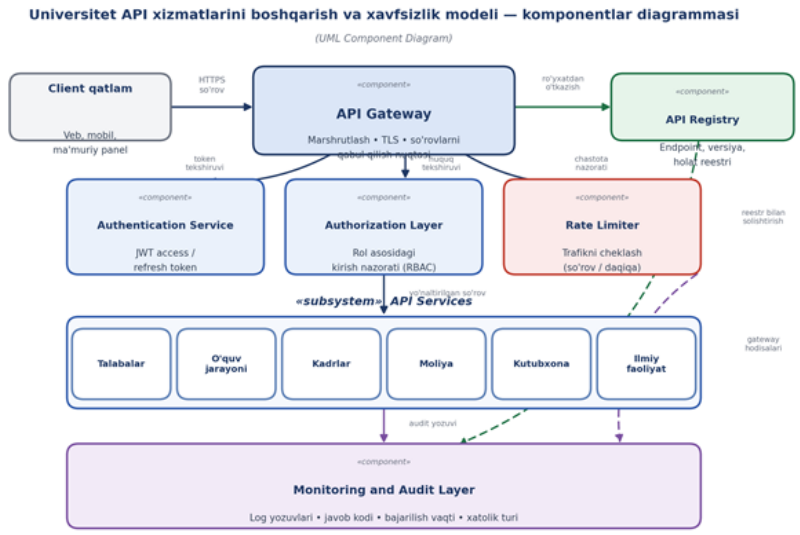
4. API trafikining nazoratsizligi. Bitta foydalanuvchi yoki dastur tomonidan juda ko‘p so‘rov yuborilishi server resurslarining ortiqcha sarflanishiga, xizmat sifatining pasayishiga va hatto xizmatning to‘xtab qolishiga olib kelishi mumkin. OWASP API Security Top 10 ushbu holatni “Unrestricted Resource Consumption” sifatida alohida xavf kategoriyasiga kiritadi [4].

5. API versiyalarini boshqarish muammosi. Eski API versiyalari uzoq vaqt faol qolishi, yangi xavfsizlik siyosatlariga moslashtirilmasligi va nazoratsiz endpointlarning mavjud bo‘lishiga sabab bo‘lishi mumkin. OWASP API Security Top 10 ham API hostlari, endpointlari va versiyalarining yangilanib turadigan inventarini yuritishni muhim xavfsizlik chorasi sifatida ko‘rsatadi [4].

6. Monitoring va hodisalarga javob berish tizimining yetishmasligi. Ko‘pgina universitet axborot tizimlarida API chaqiruvlari bo‘yicha markazlashtirilgan jurnal yuritish amaliyoti mavjud emas. Bu esa nostandart faoliyatni (masalan, bir IP manzildan ko‘p miqdorda muvaffaqiyatsiz autentifikatsiya urinishlarini) o‘z vaqtida aniqlashni qiyinlashtiradi va xavfsizlik hodisasi yuz berganda uning sabablarini tahlil qilish imkoniyatini cheklaydi.

Taklif etilayotgan API boshqaruv va xavfsizlik modeli

Tadqiqot natijasida universitet axborot tizimlari uchun markazlashtirilgan boshqaruv modeli taklif etiladi; arxitektura 1-rasmda UML komponentlar diagrammasi shaklida keltirilgan.



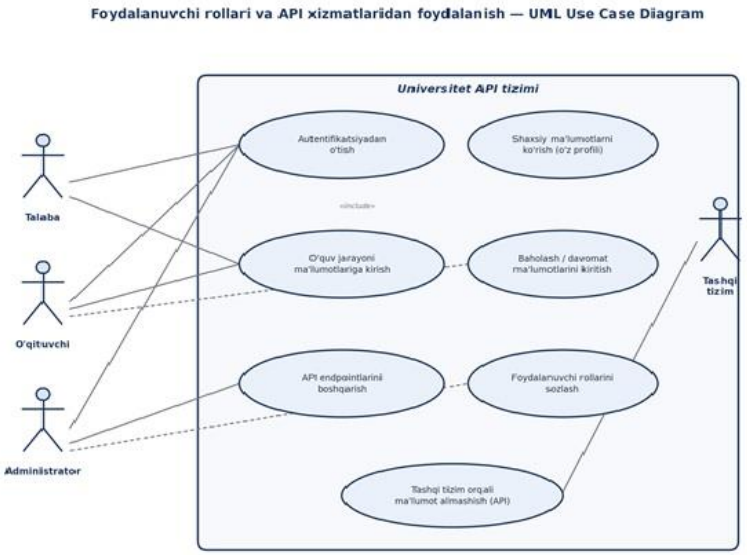
1-rasm. Universitet API xizmatlarini boshqarish va xavfsizlik modelining komponentlar diagrammasi

Model quyidagi asosiy komponentlardan tashkil topadi. 1) Client qatlam — veb-ilovalar, mobil ilovalar va ma’muriy panellar API xizmatlarining iste’molchilari hisoblanadi; bu qatlam foydalanuvchi interfeysini taqdim etadi, ammo hech qanday xavfsizlik qarori ushbu darajada yakuniy deb qabul qilinmaydi — barcha tekshiruvlar API Gateway va undan keyingi qatlamlarda takrorlanadi. 2) API Gateway — barcha tashqi so‘rovlar shu qatlam orqali o‘tadi; Gateway autentifikatsiya, avtorizatsiya, marshrutlash, trafikni cheklash va umumiy xavfsizlik siyosatlarini qo‘llash funksiyalarini bajaradi, yagona kirish nuqtasi sifatida ishlaganligi sababli siyosatlarni har bir modulda alohida takrorlash zarurati bartaraf etiladi [13]. 3) Authentication Service — foydalanuvchi identifikatsiyasi va tokenlarni boshqarish shu xizmat orqali amalga oshiriladi; Django REST Framework asosidagi arxitekturada JWT access va refresh tokenlaridan foydalanish amaliyoti mavjud [1, 9]. 4) Authorization Layer — foydalanuvchining roli va huquqlariga asoslanib endpointga murojaat qilish imkoniyatini tekshiradi, bu yerda rolga asoslangan va zarur hollarda obyektga asoslangan kirish nazorati qo‘llaniladi. 5) Rate Limiter — har bir foydalanuvchi, mijoz yoki IP manzil uchun ma’lum vaqt oralig‘ida ruxsat etilgan so‘rovlar sonini nazorat qiladi. 6) API Services — talabalar, o‘quv jarayoni, kadrlar, moliya, kutubxona va ilmiy faoliyat kabi xizmatlar shu qatlamda joylashadi; har bir xizmat o‘z biznes

mantig‘ini mustaqil bajaradi, ammo kirish xavfsizligi bo‘yicha qaror qabul qilmaydi. 7) Monitoring and Audit Layer — so‘rovlarning vaqti, endpointi, foydalanuvchisi, javob kodi va bajarilish holati qayd etiladi. 8) API Registry — faol endpointlar, versiyalar, xizmat egasi, huquq talablari va holati haqidagi ma’lumotlarni saqlovchi markazlashtirilgan reestr; Registry Gateway bilan uzviy bog‘liq holda ishlaydi va yangi endpoint qo‘shilganda avtomatik yangilanishi lozim.

Foydalanuvchi rollari va foydalanish holatlari tahlili

Asosiy rollar va ular bajaradigan operatsiyalar 2-rasmda UML use case diagrammasi orqali tasvirlangan. Talaba faqat autentifikatsiyadan o‘tib o‘z ma’lumotlarini ko‘rish huquqiga ega; o‘qituvchi baholash va davomat ma’lumotlarini kiritish huquqiga ega; administrator endpointlarni boshqarish va rollarni sozlash huquqiga ega; tashqi tizimlar (masalan, Vazirlik axborot tizimlari) alohida aktyor sifatida bir xil xavfsizlik siyosatlariga bo‘ysunadi.



2-rasm. Foydalanuvchi rollari va API xizmatlaridan foydalanish holati

API xavfsizligini ta’minlash mexanizmlari

Autentifikatsiya. Muvaffaqiyatli autentifikatsiyadan so‘ng access token beriladi. JWT asosidagi mexanizm universitetning mavjud tizimlariga mos keladi [9]; Laravel backend Authorization: Bearer tokenini qabul qiladi [1]. RFC 6750 bearer tokenlarni

himoyalashda TLS talab qiladi [6]; access token muddati qisqa (15–30 daqiqa), refresh token esa HttpOnly cookie orqali saqlanishi tavsiya etiladi [9].

Avtorizatsiya. Gateway yoki backend darajasida foydalanuvchi roli, modulga kirish huquqi, HTTP metodiga ruxsat, obyektga kirish huquqi va administrator funksiyalaridan foydalanish huquqi tekshiriladi — bu OWASP bo‘yicha obyekt va funksiya darajasidagi avtorizatsiya muammolarining oldini oladi [4].

Rate Limiting. Gateway har bir foydalanuvchi/mijoz uchun vaqt birligida ruxsat etilgan so‘rovlar sonini nazorat qiladi. Barcha endpointlar uchun bir xil emas, balki ahamiyatiga qarab differensial limit tavsiya etiladi (1-jadval).

Endpoint turi	Tavsiya etilgan limit	Asoslanish
Autentifikatsiya (login)	5–10 so‘rov / daqiqa	Brute-force hujumlarining oldini olish
O‘qish (GET)	100–300 so‘rov / daqiqa	Oddiy foydalanuvchi faoliyatiga mos
Yozish (POST/PUT/DELETE)	20–50 so‘rov / daqiqa	Ma’lumotlar yaxlitligini himoyalash
Tashqi integratsiya API’lari	SLA asosida kelishilgan	Shartnoma talablari

1-jadval. Endpoint turlariga qarab differensial rate limiting namunasi

API inventarizatsiyasi. Har bir API quyidagi atributlar asosida reestrga kiritiladi (2-jadval), bu nazoratsiz yoki eskirgan endpointlarni aniqlash imkonini beradi.

Atribut	Mazmuni
API nomi / Endpoint / Versiya	Xizmat nomi, manzili, v1/v2...

HTTP metod / Modul	GET, POST, PUT, DELETE; tegishli tizim
Autentifikatsiya / Rol	Talab qilinishi va kim foydalanishi mumkinligi
Holat / Mas’ul xizmat	Active/Deprecated; API egasi

2-jadval. API Registry uchun tavsiya etilgan atributlar tarkibi

Logging va monitoring. Har bir chaqiruv bo‘yicha vaqt, endpoint, HTTP metod, foydalanuvchi identifikatori, IP manzil, javob kodi, bajarilish vaqti va xatolik turi qayd etiladi; bu jurnallashtirish, IDS/IPS va zaxira nusxalash bilan birga xavfsizlik qatlamini tashkil etadi [7].

Transport xavfsizligi. Barcha so‘rovlar TLS 1.2+ (tavsiyan TLS 1.3) orqali uzatiladi [14], bu RFC 6750 talabiga muvofiq tokenlarning ushlab qolinishining oldini oladi [6]. Ichki mikroservislar orasida mTLS tavsiya etiladi [10], nozik ma’lumotlar esa ISO/IEC 27001 talablariga muvofiq shifrlangan holda saqlanadi [11].

OWASP API Security Top 10 talablari bilan moslik tahlili

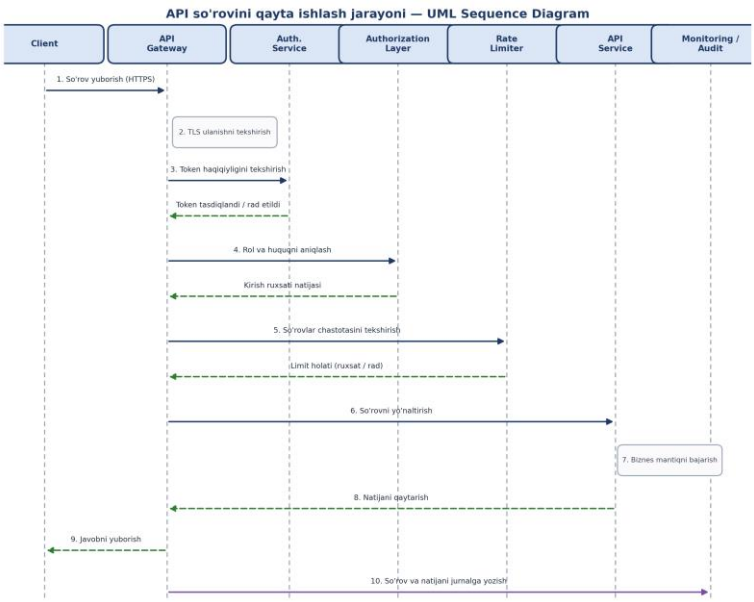
OWASP xavf toifasi	Modeldagi tegishli chora
Obyekt darajasidagi avtorizatsiya buzilishi	Authorization Layerda obyektga asoslangan nazorat
Noto‘g‘ri autentifikatsiya	Markazlashtirilgan Authentication Service, JWT
Resurslardan cheklanmagan foydalanish	Rate Limiter, differensial limitlar
Funksiya darajasidagi avtorizatsiya buzilishi	Rolga asoslangan kirish nazorati (RBAC)
Xavfsizlik konfiguratsiyasining noto‘g‘ri tashkil etilishi	Markazlashtirilgan sozlamalar (API Gateway)

API inventarizatsiyasining yetarli emasligi	API Registry — endpoint va versiyalar reestri
Xavfsiz bo‘lmagan uchinchi tomon API’laridan foydalanish	Tashqi integratsiyalar uchun SLA va monitoring

3-jadval. Taklif etilayotgan modelning OWASP API Security Top 10 (2023) talablari bilan moslik tahlili

Taklif etilayotgan modelning ishlash algoritmi

API so‘rovining qayta ishlanish jarayoni 3-rasmda UML ketma-ketlik (sequence) diagrammasi orqali tasvirlangan: (1) client so‘rov yuboradi; (2) Gateway so‘rovni qabul qiladi; (3) TLS orqali xavfsiz aloqa ta’minlanadi; (4) token haqiqiyligi tekshiriladi; (5) foydalanuvchi roli va huquqi aniqlanadi; (6) rate limiting tekshiriladi; (7) so‘rov tegishli xizmatga yo‘naltiriladi; (8) xizmat biznes mantiqini bajaradi; (9) natija clientga qaytariladi; (10) so‘rov va natija audit jurnaliga yoziladi. Har bir bosqichda salbiy natija (yaroqsiz token, limit oshishi) so‘rovni keyingi bosqichga o‘tkazmaydi va xatolik audit jurnaliga yozib qo‘yiladi.



3-rasm. API so‘rovini qayta ishlash jarayoni

Modelni amalga oshirish bosqichlari va baholash mezonlari

Bosqich	Asosiy vazifalar	Kutilayotgan natija
1. Tayyorgarlik	Mavjud APIlarni aniqlash va inventarizatsiya qilish	Boshlang‘ich API Registry
2. Markazlashtirish	Gatewayni joriy etish, endpointlarni marshrutlash	Yagona kirish nuqtasi
3. Xavfsizlikni kuchaytirish	Autentifikatsiya, avtorizatsiya, rate limiting siyosatlari	OWASP asosiy xavflari qoplanadi
4. Monitoring va takomillashtirish	Audit jurnali, metrikalar, siyosatlarni sozlash	Uzluksiz xavfsizlik nazorati

4-jadval. Modelni amalga oshirishning bosqichma-bosqich rejasi

Modelning amaliy samaradorligini baholash uchun quyidagi mezonlar tavsiya etiladi: autentifikatsiya/avtorizatsiya jarayonining o‘rtacha kechikish vaqti; Gatewayning yuklama ostidagi barqarorligi; rate limiting mexanizmining samaradorligi; API Registry orqali aniqlangan eskirgan endpointlar soni; xavfsizlik hodisalarini aniqlash vaqti (MTTD); hodisani audit orqali tahlil qilish vaqti.

Natijalar va muhokama

Taklif etilgan modelning asosiy ilmiy-amaliy natijasi API xizmatlarini oddiy ma’lumot almashish interfeysi sifatida emas, balki universitet raqamli infratuzilmasining mustaqil boshqaruv obyekti sifatida tashkil etishdan iborat. Mavjud universitet tizimlarida REST API va JWT asosidagi autentifikatsiya allaqachon amaliy qo‘llanilayotganligi ushbu yondashuvni kengaytirish uchun texnologik asos mavjudligini ko‘rsatadi [1, 9]. Taklif etilayotgan modelning avvalgi integratsiya modelidan asosiy farqi shundaki, bunda markaziy masala modullarni qanday

birlashtirish emas, balki ular taqdim etayotgan API xizmatlarini qanday boshqarish va himoyalash masalasidir.

Model quyidagi natijalarni ta’minlashga yo‘naltirilgan: API xizmatlarining yagona reestrini shakllantirish; autentifikatsiya mexanizmlarini markazlashtirish; rollarga asoslangan avtorizatsiyani qo‘llash; API trafikini nazorat qilish; eskirgan endpointlarni aniqlash; xavfsizlik hodisalarini audit qilish; API xizmatlarining ishlash holatini monitoring qilish; universitet axborot tizimlari uchun yagona API xavfsizlik siyosatini shakllantirish. OWASP API Security Top 10 ma’lumotlariga ko‘ra, API xavfsizligida avtorizatsiya, autentifikatsiya, resurslardan cheklanmagan foydalanish, noto‘g‘ri konfiguratsiya va API inventarizatsiyasi kabi masalalar muhim xavf yo‘nalishlarini tashkil qiladi [4]. Shu sababli taklif qilinayotgan modelda xavfsizlik faqat token tekshirish bilan cheklanmasdan, API hayotiy siklining bir nechta bosqichini qamrab oladi.

Shu bilan birga, modelning UML diagrammalari orqali rasmiylashtirilishi uni amaliyotchilar (dasturchilar, tizim administratorlari) uchun tushunarli va joriy etishga tayyor holatga keltiradi, chunki komponentlar, ma’lumot oqimi va foydalanuvchi rollari orasidagi munosabatlar vizual tarzda aniq ko‘rsatilgan [12].

Xulosa: Tadqiqotda universitet axborot tizimlarida API xizmatlarini boshqarish va xavfsizligini ta’minlashga qaratilgan markazlashtirilgan model taklif etildi. Model API Gateway, autentifikatsiya xizmati, avtorizatsiya qatlami, API xizmatlari, monitoring va audit hamda API reestridan tashkil topadi.

Taklif etilgan yondashuvda API xavfsizligi faqat foydalanuvchini autentifikatsiya qilish bilan chegaralanmaydi. Endpointlarga kirish huquqlarini nazorat qilish, trafikni cheklash, API versiyalarini boshqarish, xizmatlar inventarini yuritish, jurnal yozuvlarini shakllantirish va monitoring qilish yagona boshqaruv konsepsiyasiga birlashtiriladi. Universitetning mavjud axborot tizimlarida REST API va JWT asosidagi autentifikatsiya mexanizmlaridan foydalanilayotganligi [1] taklif etilayotgan modelni mavjud infratuzilmaga bosqichma-bosqich joriy etish imkoniyatini beradi.

Kelgusidagi tadqiqotlarda ushbu modelning amaliy prototipini ishlab chiqish, API endpointlarining ishlash tezligi, autentifikatsiya kechikishi, rate limiting samaradorligi, xavfsizlik hodisalarini aniqlash darajasi va tizimning yuklama ostidagi barqarorligini tajriba asosida baholash maqsadga muvofiq.

Foydalanilgan adabiyotlar

1. Istamov B.B. Universitet dasturiy modullarini yagona axborot tizimiga integratsiyalashning arxitekturaviy modeli va samaradorlik ko‘rsatkichlari. Journal of Future Science, Technology and Innovation, 2026.
2. Xusenov M.Z., Istamov B.B. Universitet axborot tizimlarini integratsiyalashda kutubxona modulining arxitektura va funksional yechimlari. Scientific Reports of Bukhara State University, 2026.
3. Xusenov M.Z., Istamov B.B. Universitet axborot tizimlarida ma’lumotlar sinxronizatsiyasiga asoslangan modul integratsiyasi modeli. Ilm-fan va texnologiyalar, 2026.
4. OWASP Foundation. OWASP API Security Top 10 – 2023. API Security Project.
5. OWASP Foundation. OWASP API Security Project. API xavfsizligi va zaifliklarni boshqarish bo‘yicha metodik materiallar.
6. Jones M., Hardt D. The OAuth 2.0 Authorization Framework: Bearer Token Usage. RFC 6750. IETF, 2012.
7. Xusenov M.Z., Istamov B.B. Universitet kutubxonalarida elektron resurslar boshqaruv tizimi. Innovatech: Conference on Scientific Innovations and Research.
8. Fielding R.T. Architectural Styles and the Design of Network-based Software Architectures. PhD dissertation, UC Irvine, 2000.
9. Jones M., Bradley J., Sakimura N. JSON Web Token (JWT). RFC 7519. IETF, 2015.

10. NIST. Security Strategies for Microservices-based Application Systems. NIST SP 800-204, 2020.
11. ISO/IEC 27001:2022 – Information security, cybersecurity and privacy protection – ISMS – Requirements.
12. OMG. OMG Unified Modeling Language (UML), Version 2.5.1. Specification, 2017.
13. Richardson C. Microservices Patterns: With Examples in Java. Manning Publications, 2018.
14. Rescorla E. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446. IETF, 2018.