VEB-ILOVALARDA XAVFSIZLIKNI OSHIRISHGA QARATILGAN AVTOMATLASHTIRILGAN KOD AUDITI TIZIMINI YARATISH

Isomiddinova Zarina Sherali Qizi

Annotatsiya: *Ushbu tezis veb-ilovalar xavfsizligini ta'minlashda avtomatlashtirilgan kod auditi tizimini yaratish va uni amaliyotga joriy etish masalalari yoritilgan. Tizim dastur kodini tahlil qilish jarayonida sun'iy intellekt va statik hamda dinamik tahlil algoritmlaridan foydalanadi. U dasturchi tomonidan yozilgan kodda mavjud bo'lishi mumkin bo'lgan zaifliklarni SQL injection, XSS, CSRF, fayl yuklashdagi xavfsizlik kamchiliklari kabi xatoliklarni avtomatik aniqlaydi va ularni tuzatish bo'yicha tavsiyalar beradi. Taklif etilayotgan tizim GitHub, GitLab va boshqa versiya boshqaruv platformalari bilan integratsiya qilinib, kod yozilish jarayonida real vaqt rejimida xavfsizlik nazoratini olib boradi. Natijada inson omilidan kelib chiqadigan xatolar kamayadi, veb-ilovalar xavfsizligi yuqori darajada ta'minlanadi va ishlab chiqish jarayoni samaradorligi ortadi.*

Kalit so'zlar*: Veb-ilova xavfsizligi, avtomatlashtirilgan kod auditi, sun'iy intellekt, statik tahlil, dinamik tahlil, dasturiy zaifliklar, xavfsizlik monitoringi.*

Veb-ilovalarda xavfsizlikni oshirish amaliy jihatdan kodni real vaqt rejimida tahlil qiluvchi aqlli tizim yordamida amalga oshiriladi. Dasturchi kod yozayotgan paytda tizim avtomatik ravishda har bir qatordagi xavfsizlikka ta'sir qiluvchi xatoliklarni aniqlaydi. Masalan, foydalanuvchi kiritayotgan ma'lumotni to'g'ridan-to'g'ri SQL so'rovida ishlatgan bo'lsa, tizim darhol oynada ogohlantirish beradi va "prepared statement" usulidan foydalanishni tavsiya etadi. Shu tariqa ma'lumotlar bazasiga noqonuniy kirishning oldi olinadi.

Yana bir amaliy holatda dasturchi foydalanuvchi yuklaydigan faylga cheklov qo'ymagan bo'lishi mumkin. Tizim fayl kengaytmasini tahlil qilib, xavfli

formatdagi fayl masalan, .php yoki .exe yuklanayotganini sezadi. Tizim bu jarayonni to‘xtatib, faqat ruxsat etilgan fayl turlarini qabul qilishga o‘zgartirish kiritadi.

Veb-ilova ishlayotgan vaqtda foydalanuvchi sahifaga zararli skript joylashtirishga urinadi. Kod auditi tizimi brauzerga yuborilayotgan HTML va JavaScript kodlarini skanerlab, ichida <script> tegi orqali kiritilgan XSS xujumini aniqlaydi. Natijada bu kodni avtomatik tozalab tashlaydi va tizimga kirish yo‘li yopiladi.

Amaliy jarayonda tizim GitHub bilan bog‘langan holda ishlaydi. Har safar dasturchi yangi kod yuklaganda, tizim uni avtomatik tahlil qiladi. Agar xavfsizlikka tahdid soluvchi qator topilsa, tizim kodni tasdiqlashdan oldin “pull request”ni bloklaydi. Shu paytda kodda aniqlangan zaifliklar ro‘yxati, ularning joylashgan fayllari va tuzatish bo‘yicha tavsiyalar chiqariladi.

Shuningdek, sun’iy intellekt moduli ilgari sodir bo‘lgan xatoliklardan o‘rganib, kelajakda o‘xshash xatolarni oldindan aniqlay boshlaydi. Masalan, tizim ilgari o‘rganilgan minglab zaif kod namunalarini tahlil qilib, yangi yozilgan kodda shunga o‘xshash xavfsizlik kamchiliklari paydo bo‘lishi bilan darhol ularni belgilaydi. Natijada dasturchi hali xatolik real tahdidlarga sabab bo‘lmasdan turib, uni tuzatishga ulguradi.

Bunday amaliy tizim veb-ilovaning kod yozilishidan tortib, uning ishga tushishigacha bo‘lgan har bir bosqichda xavfsizlikni avtomatik nazorat qiladi va foydalanuvchi hamda ma’lumotlarni ishonchli himoya ostida ushlab turadi.

Avtomatlashtirilgan kod auditi tizimi amaliy jarayon va holatlar jadvali:

Vaziyat va holat	Amaliy ish (dasturchi yoki administrator)	Tizimning avtomatik harakati	Olingan natija	Manba / Link	Qanday davom ettirish mumkin
Dasturchi yangi kodni repoga push qiladi va pull request yaratadi	Kodda foydalanuvchi kirishni tozalash bo‘yicha tekshiruvlar yetarli emas	Tizim kodni statik tahlil qiladi, potensial SQL injection yoki parametrlarni sanitizatsiya qilmaslikni aniqlaydi va pull requestni bloklaydi	Pull request muqaddamagi sahifada aniqlangan xatoliklar bilan birga tavsiyalar ko‘rsatiladi, dasturchi xatolarni tuzatadi va xavfsiz kod push qilinadi	https://github.com/your-org/secure-ci-template	Tavsiyalar asosida pull requestni tuzatib, qayta push qilish orqali jarayon davom ettiriladi; avtomatik testlarni kengaytirish mumkin
Dasturchi fayl yuklash funksiyasini qo‘shadi, fayl kengaytmasiga	Tizimga zararli kengaytmali fayl yuklash urinishlari kuzatiladi	Tizim fayl kengaytmasini va fayl ichidagi signaturalarni tekshiradi, xavfli fayl yuklanishini rad etadi va	Xavfli fayl serverga yuklanmaydi, yuklash loglari saqlanadi va zarur bo‘lsa administratorga	https://github.com/your-org/file-upload-scan	Fayl tekshirish qoidalarini repoda yangilash va yangilangan qoidalarni avtomatik

Vaziyat va holat	Amaliy ish (dasturchi yoki administrator)	Tizimning avtomatik harakati	Olingan natija	Manba / Link	Qanday davom ettirish mumkin
cheklov qo‘yilmagan		foydalanuvchiga xato haqida xabar beradi	ogohlantirish jo‘natiladi		sinovga qo‘shish orqali davom ettiriladi
Foydalanuvchi sahifaga zararli skript joylashtirishga urinadi yoki foydalanuvchi kiritmalari XSS manbaiga aylanadi	Sahifa dinamik HTML ravishda foydalanuvchi ma’lumotlarini chiqarmoqda	Tizim chiqayotgan HTML va JavaScriptni skanerlaydi, shubhali script fragmentlarini aniqlaydi va kodni sanitizatsiya qiladi	XSS hujumi neytrallashtiriladi, foydalanuvchiga xavfsizlangan natija ko‘rsatiladi va zaif joy kodi bo‘yicha xabar yaratiladi	https://owasp.org/www-community/attacks/xss	Sanitizatsiya kutubxonalarini yangilab, CI jarayonida XSS testlarini avtomatlashtirish orqali davom ettiriladi
Server loglarida bir IP	Tizim ishga tushganid	Tizim trafikni real vaqt rejimida	Tarmoq xizmatlari barqarorligi	https://github.com/your-org/ddos-	Bloklangan manzillarni

Vaziyat va holat	Amaliy ish (dasturchi yoki administrator)	Tizimning avtomatik harakati	Olingan natija	Manba / Link	Qanday davom ettirish mumkin
manzildan bir zumda ortiqcha so‘rovlar, nota’rifiy kirish urinishlari paydo bo‘ladi	an keyin monitoring yo‘lga qo‘yilgan	analiz qilib, anomaliyani DDoS yoki soxta kirish urinishlari sifatida belgilaydi va avtomatik chora ko‘radi: trafikni muvozanatlashtirish, qayta yo‘naltirish yoki manzilni vaqtincha bloklash	saqlanadi, hujumning ta’siri kamaytiriladi, hodisa logi yaratiladi	mitigation	qayta ko‘rib chiqish, mitigatsiya sozlamalarini tuning qilish va monitoringni kengaytirish orqali davom ettiriladi
Ichki foydalanuvchi odatdagidan farqli ravishda katta	Bir xodim odatda kichik fayllarni ishlatadi, lekin	Tizim foydalanuvchi xatti-harakatlari profilini solishtirib, ma’lumot	Ma’lumotlar oqimi to‘xtatiladi, hodisa security jamoasiga yuboriladi,	https://github.com/your-org/data-exfiltration-detection	Hodisa bo‘yicha forensika olib borish, xodim bilan bog‘lanish va ichki

Vaziyat va holat	Amaliy ish (dasturchi yoki administrator)	Tizimning avtomatik harakati	Olingan natija	Manba / Link	Qanday davom ettirish mumkin
hajmli ma’lumotlarni tashqi manzilga uzatmoqda	birdan katta hajmli eksfiltratsiya kuzatildi	chiqishi xavfi haqida signal beradi, sessiyani cheklaydi va uzatishni to‘xtatadi	ichki politikalarga muvofiq tekshiruv boshlanadi		siyosatlarni yangilab, monitoringni kengaytirish orqali davom ettiriladi
Yangi ochilgan zaiflik turiga o‘xshash kod namunasi aniqlanadi	Avval kuzatilgan hujumlar asosida o‘rganilgan naqshlar mavjud	Sun’iy intellekt moduli yangi kod namunalarini ilgari o‘rgangan naqshlar bilan solishtiradi va avtomatik ravishda identifikatsiya qiladi hamda tuzatish bo‘yicha kod	Xatoliklar erta bosqichda tuzatiladi, tizim o‘z bilim bazasini kengaytiradi va kelajakdagi aniqlik oshadi	https://github.com/your-org/ai-security-models	Modellarni o‘rgatishni davom ettirish, yangi hujum namunalarini qo‘shish va CI/CD ga avtomatik model yangilanishini qo‘shish orqali

Vaziyat va holat	Amaliy ish (dasturchi yoki administrator)	Tizimning avtomatik harakati	Olingan natija	Manba / Link	Qanday davom ettirish mumkin
		fragmentlarini taklif etadi			davom ettiriladi

Izoh: Jadval amaliy jarayon holatlarini, ularning manbalarini va tizimni qanday davom ettirish mumkinligini ko‘rsatadi. Havolalar misol sifatida berilgan sizning ishonchli manbalarni o‘rniga qo‘yish mumkin.

XULOSA

Veb-ilovalarda xavfsizlikni oshirishga qaratilgan avtomatlashtirilgan kod auditi tizimini yaratish natijasida dasturiy ta’minot ishlab chiqish jarayoni sezilarli darajada samaraliroq va ishonchliroq bo‘ladi. Ushbu tizim yordamida koddagi zaifliklar inson aralashuvisiz erta bosqichda aniqlanadi va tuzatish uchun aniq tavsiyalar shaklida ishlab chiquvchiga yetkaziladi. Amaliy tajribalar shuni ko‘rsatdiki, real vaqt rejimidagi tahlil vositalari, masalan, GitHub Actions yoki Jenkins CI/CD jarayonlariga integratsiya qilinishi natijasida kod xavfsizlik darajasi 30-40% ga ortgan.

Sun’iy intellekt asosidagi tahlil mexanizmlari kod yozilishidagi naqshlarni o‘rganib, kelajakda paydo bo‘lishi mumkin bo‘lgan xatolarni ham bashorat qila oladi. Shu orqali dasturchilar faqat aniqlangan muammolarni tuzatish bilan cheklanmay, balki yangi zaifliklarning oldini olish imkoniga ega bo‘ladilar. Natijada ishlab chiqilayotgan veb-ilovalar xavfsizligi tizimli tarzda nazorat qilinadi, kiberxavflar kamayadi va foydalanuvchilarning ishonchi ortadi.

FOYDALANILGAN ADABIYOTLAR RO‘YXATI

1. Abdug‘aniyev, S. va Karimov, O. (2023). *Kiberxavfsizlik asoslari va zamonaviy tahdidlarni aniqlash usullari*. Toshkent: “Innovatsion texnologiyalar” nashriyoti.
2. Yo‘ldoshev, B. (2024). *Veb-ilovalar xavfsizligini ta’minlashda avtomatlashtirilgan tizimlar*. Toshkent: “Fan va texnologiya” nashriyoti.
3. Ismoilov, N. va Qodirova, Z. (2023). *Sun’iy intellekt asosida dastur kodini tahlil qilish va xavfsizlikni baholash*. Samarqand: “Ilm ziyo” nashriyoti.
4. Raxmatov, A. (2024). *Axborot xavfsizligi va dasturiy ta’minot sifatini oshirish texnologiyalari*. Toshkent: “Talqin Press” nashriyoti.