Web Segurtasuna

Mikel Egaña Aranguren

mikel-egana-aranguren.github.io

mikel.egana@ehu.eus

Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

Web Segurtasuna

https://doi.org/10.5281/zenodo.4302267

https://github.com/mikel-egana-aranguren/EHU-SGSSI-01

Pen-testing

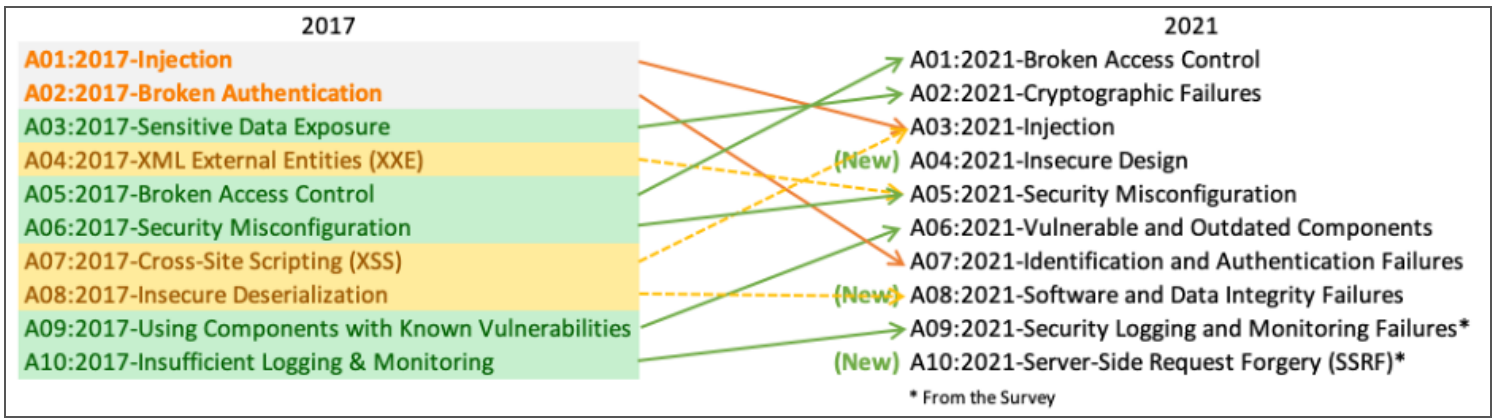
Penetration testing: sistema batean erasotzaile potentzial baten tresna berberak erabiliz sartzen saiatzea, ahuleziak aurkitzeko eta konpontzeko

Urrakortasun nagusiak Web Sistemetan

Open Web Application Security Project (OWASP) erakundeak urrakortasun ohikoenak aztertzen ditu

aldizkako txostena: OWASP Top Ten (~~2017~~, 2021)

Urrakortasun nagusiak Web Sistemetan



Common Weakness Enumerations (CWEs)

https://cwe.mitre.org/

Urrakortasun nagusiak Web Sistemetan

- Sarbide-kontrola apurtzea
- Akats kriptografikoak
- Injekzioa
- Diseinu insegurua
- Segurtasun-konfigurazio eskasa
- Osagai kalteberak eta zaharkituak

Urrakortasun nagusiak Web Sistemetan

- Identifikazio- eta autentifikazio-akatsak
- Akatsak datuen eta softwarearen osotasunean
- Akatsak segurtasunaren monitorizazioan

Sarbide-kontrola apurtzea

Broken access control (A01:2021)

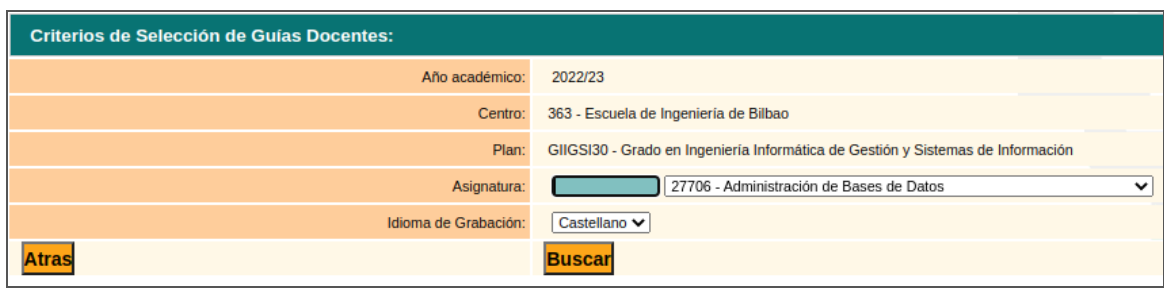
Sarbide-kontrola apurtzea

Sarbide-kontrolaren ondorioz, definitutako baimenek ezarritako mugen barruan bakarrik jardun behar du erabiltzaileak

Sarbide-kontrolean huts egiteak ondorio larriak dakartza

Sarbide-kontrola apurtzea: ahuleziak

"Ukatzea lehenetsia" printzipioa urratzea: sarbidea gaitasun, rol edo erabiltzaile jakin batzuei bakarrik bermatu beharko litzaieke, baina edozeinen esku dago



Sarbide-kontrola apurtzea: ahuleziak

Sarbide-kontrola URL parametroen edo API eskaeren bidez saihestea

Kontu pertsonal batera sartu eta datuak aldatu erabiltzailearen identifikatzailearekin

API sarbidea kontrolik gabe HTTP POST, PUT, eta DELETE metodoetarako

Sarbide-kontrola apurtzea: ahuleziak

Pribilegioa igotzea: administratzaile gisa jardutea, erabiltzaile gisa logeatuta egonik

Replaying edo JSON Web Token (JWT) aldatzea pribilegioak igotzeko

CORSen konfigurazio okerrak baimendu gabeko jatorrietatik APIra sartzeko aukera ematen du

Sarbide-kontrola apurtzea: ahuleziak

- CWE-200: Exposure of Sensitive Information to an Unauthorized Actor
- CWE-201: Insertion of Sensitive Information Into Sent Data
- CWE-352: Cross-Site Request Forgery (CSRF)
- …

Sarbide-kontrola apurtzea: prebentzioa

Lehenetsitako ukapena baliabide guztietarako, publikoak izan ezik

Kontrolerako sarbide-mekanismoak behin bakarrik ezartzea eta aplikazio osora zabaltzea, CORSen erabilera minimizatuz

Direktorioen zerrendaketa ez baimentzea eta aplikazioaren root mailan metadatuen fitxategirik (adibidez, Git) edo backup-ik ez dagoela ziurtatzea

Sarbide-kontrola apurtzea: prebentzioa

Sartzeko ahalegin guztiak logeatzea, administratzaileak ohartaraztea beharrezkoa denean (adibidez, huts egindako sarbide asko)

APIetan eskaeren tasa mugatzea, programen bidezko erasoak saihesteko

Sarbide-kontrola apurtzea: prebentzioa

Saioaren identifikatzaileak:

- Zerbitzariarena (stateful): baliogabetzea deslogeatzean
- Stateless JWT tokens: oso bizitza laburra
- Denbora luzeagoko JWTrentzat, jarraitu OAuth protokoloa sarbidea baliogabetzeko

Sarbide-kontrola apurtzea: adibidea

SQL deia kontuko informazioa lortzeko, egiaztatu gabeko datuekin:

```
01. pstmt.setString(1, request.getParameter("acct"));
02. ResultSet results = pstmt.executeQuery( );
```

Erasotzaileak URL hau ikusi behar du soilik:

https://example.com/app/accountInfo?acct=notmyacct

Akats kriptografikoak

Cryptographic Failures (A02:2021)

Akats kriptografikoak

Erabilitako metodo kriptografikoen akatsak, ondorioz datuak azaltzen direlarik

Datuak babesteko beharrak zehaztea, bai transmisioan, bai biltegiratzean (At rest): datuak babesteko legeak

Akats kriptografikoak

Datuak testu lau gisa transmititzen dira? Kanpoko zein barruko konexioetan (adibidez, karga-balantzearen zerbitzariak): HTTP, SMTP, FTP ...

Algoritmo edo protokolo zaharkituak edo ahulak erabiltzen dira?

Gako ahulak edo gako lehenetsiak erabiltzen dira?

Gakoen kudeaketa egokia erabiltzen da?

Gakoen kudeaketa

Gako baten bizi-zikloa:

1. Sorkuntza
2. Banaketa
3. Erabilpena
4. Biltegiratzea
5. Errotazioa, ezeztapena, suntsipena

Gakoen kudeaketa: Sorkuntza

Algoritmo ahulak ekidin

Key Generator edo entropia altuko Random Number Generator -ak erabili

Gakoen kudeaketa: Banaketa

Man in the middle erasoak ekiditeko SSL/TLS bezalako konexio seguruak erabili

Gakoen kudeaketa: Erabilpena

Baimendutako pertsonek soilik erabiltzen dituztela gakoak bermatu

Gakoen kudeaketa: Biltegiratzea

Egokiena hardware dedikatua erabiltzea: Hardware Security Module:

- Funtzionatzeari uzten diote sarbide fisiko bat dagoenean
- Estandar zorrotzak betetzen dituzte: FIPS 140-2 (4 maila)

Hodeian ere zerbitzuak erabili daitezke: Google Cloud Key Management

Gakoen kudeaketa: errotazioa

Gako batek bere balio-aldia amaitzen duenean, kendu egiten da, datuak deszifratu eta berriro zifratzen direlarik gako berri batekin

Zenbat eta denbora gehiago egon gakoren bat zirkulazioan, orduan eta arriskutsuagoa izango da

Gakoen kudeaketa: ezeztapena edo suntsiketa

Gako bat konprometitua izan bada, ezeztatu egin behar da, nahiz eta baliozkotasun-epearen barruan egon

Ezeztatutako gakoa gordetzea eskatzen duten estandarrak daude, etorkizunean datuak deszifratzeko behar izanez gero (adibidez, epaiketa batean)

Gakoa erabat suntsitu: etorkizunean ezin da erabili

Gakoen kudeaketa: jardunbide egokiak

Ez idatzi gakoaren balioak software-kodean, nahiz eta segurua izan

Pribilegio txikienaren printzipioa: erabiltzaile batek benetan behar dituen gakoetarako sarbidea soilik izan beharko luke

HSM-ak erabili

Gakoen kudeaketa: jardunbide egokiak

Gakoen kudeaketa ahalik eta gehien automatizatu

Sorkuntza, Banaketa etab. pertsona desberdinen artean banatu

Gakoak zatietan banatu

Akats kriptografikoak

Zerbitzariaren ziurtagiria baliozkoa eta konfiantzazkoa da, eta CA batek bermatzen du?

Erabiltzaileen pasahitzak zuzenean gako gisa erabiltzen al dira, Pasword Based Key Derivation Functions (PBKDFs) erabili beharrean?

Ausazkotasun-hazia behar bezain indartsua al da?

Akats kriptografikoak: prebentzioa

Transmititutako, prozesatutako eta biltegiratutako datuak pribatutasun-mailaren arabera sailkatu

Ahalik eta datu pribatu gutxien biltegiratzea, nahiz eta Tokenizazioa bezalako metodoak erabili

Biltegiratutako datu guztiak zifratu (Encription at rest)

Akats kriptografikoak: prebentzioa

Algoritmo eguneratuak (MD5, SHA1, … ez), gako indartsuak, eta Gakoen kudeaketa erabili

Transmititzen diren datu guztiak zifratu TLS FS bidez (TLS + Forward Secrecy)

Bezeroan datu berezientzako ez baimendu cache-en erabilpena

Akats kriptografikoak: prebentzioa

Pasahitzak hash eta gatzarekin biltegiratu

Entropia handiena duten haziak erabili

Akats kriptografikoak: adibideak

Aplikazio batek datuak enkriptatzeko datu-basearen funtzio lehenetsia erabiltzen du. Hala ere, datu horiek automatikoki desenkriptatzen dira kontsultatzean, eta sarbidea ahalbidetzen da, adibidez, SQL injekzio eraso bat badago

Akats kriptografikoak: adibideak

Web orrialde batek ez du TLS erabilpena derrigortzen. Erasotzaile batek trafikoa monitorizatzen badu eta konexio bat HTTPStik HTTPera "jaisten" badu, kontsultak eten eta erabiltzailearen saioko cookiea lor dezake. Erasotzaileak cookiea (replay) birbidaltzen du erabiltzailearen saioa bahituz eta erabiltzaile-datuetarako sarbidea lortuz

Datu baseak gatz gabeko hash-ak erabiltzen ditu. Erasotzaile batek hash-ak prekalkulatu ditzake GPU-ak erabiliz

Akats kriptografikoak: ahuleziak

- CWE-259: Use of Hard-coded Password
- CWE-327: Broken or Risky Crypto Algorithm
- CWE-331: Insufficient Entropy
- ...

Injekzioa

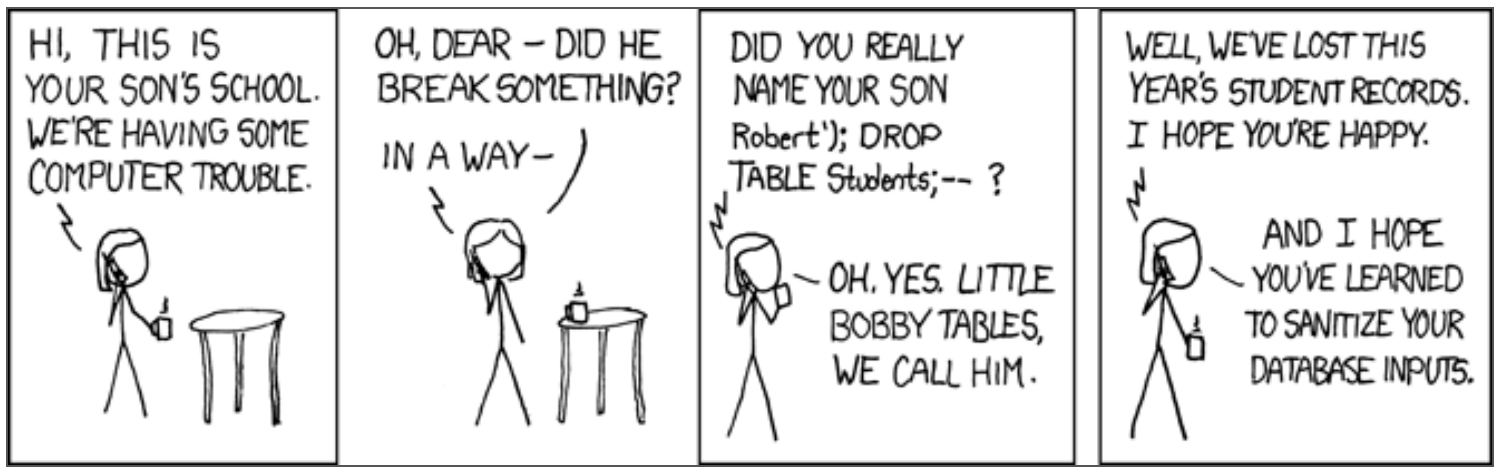
Injection (A03:2021)

Injekzioa

Aplikazio ahula da baldin eta:

- Erabiltzaileak ematen dituen datuak ez dira balioztatzen, iragazten edo garbitzen
- Interpreteak zuzenean kontsulta dinamikoak edo parametrizatu gabeko deiak erabiltzen ditu, "escaping" ik gabe
- Erabiltzaileak sartutako datu okerrak zuzenean erabiltzen dira. SQL sententziak edo erabiltzen diren komandoak egitura/datu maltzurrak ditu kontsulta dinamikoetan, komandoetan edo biltegiratutako prozeduretan

SQL injekzioa



https://xkcd.com/327/

(Injekzioa LLM-tan)

Multi-modal prompt injection image attacks against GPT-4V

SQL injekzioa

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users
WHERE UserId = " + txtUserId;
```

SQL injekzioa

Input web: `105 OR 1=1`

```
SELECT * FROM Users WHERE UserId = 105 OR 1=1;
```

OR 1 = 1 beti da true, Users zutabeko lerro guztiak itzuliko ditu

Parametrizatutako kontsultak

Parametroak: exekutatzean kontsultari gehitzen zaizkion eta hitzez hitz aztertzen diren balioak (adibidez, aldagaiak zutabe-izenak aipatzen baditu, interpreteak ziurtatzen du parametroak zutabe-izenak direla kontsultara gehitu eta exekutatu aurretik)

Parametrizatutako kontsultak

```
$stmt = $dbh->prepare("INSERT INTO Customers
(CustomerName,Address,City)
VALUES (:nam, :add, :cit)");
$stmt->bindParam(':nam', $txtNam);
$stmt->bindParam(':add', $txtAdd);
$stmt->bindParam(':cit', $txtCit);
$stmt->execute();
```

Object-Relational Mapping (ORM)

Datu-basearen edukia objektuetara itzultzen duten framework-ak

Ez dago datu-basearekin konexio zuzenik

Object-Relational Mapping (ORM)

```
String sql = "SELECT ... FROM persons WHERE id = 10"
DbCommand cmd = new DbCommand(connection, sql);
Result res = cmd.Execute();
String name = res[0]["FIRST_NAME"];
Person p = repository.GetPerson(10);
String name = p.FirstName;
```

Injekzioa: ahuleziaren detekzioa

Iturburu-kodea berrikustea

HTTP goiburuen, URL-en, Cookie-en, JSON, XML eta web input-en testatze automatizatua

Application Security Testing erabili Continous Integration/Continous Deployment prozesutan

Injekzioa: prebentzioa

Interpretea erabiltzen ez duten API seguruak erabiltzea. Ezin bada, dei parametrizatuak edo ORMak erabili

Input-ak balioztatu

Ahal diren karaktere berezi guztiak "Escape"-atu

SQLn LIMIT bezalako kontrolak erabiltzea, datuen galera masiboa saihesteko

Injekzioa

- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
- CWE-73: External Control of File Name or Path
- ...

Diseinu insegurua

Insecure Design (A04:2021)

Diseinu insegurua

Arkitektura akatsak, kodea inplementatu baino aurrekoak

Software Assurance Maturity Model

https://owaspsamm.org/

SAMM-v2-PDF.pdf

Diseinu insegurua: adibideak

e-commerce kate batek erosleen eskaera asko jaso ditzake bot-en bidez, produktuak beste leku batzuetan saltzeko. Hori ekidin daiteke, adibidez, arkitekturan bot-ak detektatzen dituzten osagaiak gehituz (Eskaera asko denbora gutxian, produktu bat argitaratzeko segundo gutxiren buruan egindako eskaera, eta abar).

Diseinu insegurua

- CWE-209 Generation of Error Message Containing Sensitive Information
- CWE-256 Unprotected Storage of Credentials
- CWE-501 Trust Boundary Violation
- CWE-522 Insufficiently Protected Credentials
- ...

Segurtasun-konfigurazio eskasa

Security Misconfiguration (A05:2021)

Segurtasun-konfigurazio eskasa

Aplikazioa zaurgarria izan liteke baldin eta:

- Hodei zerbitzu batean baimen desegokiak dauzka
- Beharrezkoak ez diren funtzioak aktibaturik daude (Portuak, zerbitzuak, orrialdeak, kontuak, baimenak)
- Lehenetsitako kontuek eta pasahitzek aktibo jarraitzen dute
- Erroreen kudeaketak informazioa azaleratzen du (adib. stack traces Java salbuespenetan) / erroreek informazio gehiegi ematen dute

Segurtasun-konfigurazio eskasa

Aplikazioa zaurgarria izan liteke baldin eta:

- Sistema eguneratuetarako, azken segurtasun-adabakiak ez daude aktibo eta/edo behar bezala konfiguratuta
- Frameworketako (Struts, Spring, ASP.net,…), liburutegietako, datu-baseetako, … segurtasun-aukerak ez daude aktibatuta
- Zerbitzariak ez du segurtasun HTTP goiburuekin erantzuten

Segurtasun-konfigurazio eskasa: prebentzioa

- Garapen automatizatuko sistema, non ingurune seguruagoa hedatzea erraza den. Garapen-, kalitate- eta ekoizpen-inguruneek konfigurazio bera izan beharko lukete, kredentzialek izan ezik. Prozesu horrek ahalik eta automatizatuena izan beharko luke, ingurune seguru berri bat ahalik eta azkarren sortzeko
- Sistema minimalista, osagai eta funtzio gehigarririk gabe

Segurtasun-konfigurazio eskasa

- CWE-16 Configuration
- CWE-611 Improper Restriction of XML External Entity Reference
- ...

Osagai kalteberak eta zaharkituak

Vulnerable and Outdated Components (A06:2021)

Osagai kalteberak eta zaharkituak

Aplikazioa zaurgarria izan liteke baldin eta:

- Ez dakizkigu osagaien bertsio zehatzak (Beti darabilgu "latest"), zuzenean erabiltzen ditugun osagaietan eta berauen dependentzietan
- Softwareak ez badu soporterik, kaltebera edo zaharkitua da: sistema eragilea, web zerbitzaria, datu-baseak kudeatzeko sistema, API-ak eta horien osagaiak, exekuzio-inguruneak eta liburutegiak

Osagai kalteberak eta zaharkituak

Aplikazioa zaurgarria izan liteke baldin eta:

- Ez baduzu zure sistema aldizka eskaneatzen urrakortasunak bilatzeko, eta ez bazaude harpidetuta erabiltzen dituzun osagaiei buruzko segurtasun-buletinetara
- Osagaiak arriskuan oinarritutako maiztasunez eguneratzen ez badituzu. Hori oso ohikoa da hilean behin adabakiak aplikatzeko politikak dituzten erakunde handietan, sistema kaltebera delarik egun batzutan

Osagai kalteberak eta zaharkituak: prebentzioa

Adabakiak aplikatzeko prozesu bat egon beharko litzateke:

- Erabiltzen ez diren dependentziak, funtzionalitateak, osagaiak, fitxategiak etab. kendu
- Zerbitzarian eta bezeroan erabiltzen diren osagaien eta beraien dependentzien zerrenda izan, automatikoki sortua. CVE bezalako baliabideak berrikusi osagaion ahultasunak jakiteko
- Osagai sinatuak soilik eskuratu, eta iturri ofizialetatik

Identifikazio- eta autentifikazio-akatsak

Identification and Authentication Failures (A07:2021)

Identifikazio- eta autentifikazio-akatsak

Erabiltzaileen nortasuna berrestea, autentifikatzea eta saioak kudeatzea funtzio kritikoak dira sistema autentifikazioan oinarritutako erasoen aurka babesteko

Identifikazio- eta autentifikazio-akatsak

Aplikazioa zaurgarria izan liteke baldin eta:

- Credential Stuffing bezalako eraso automatizatuak ezin ditu ekidin
- Pasahitz ahulak edo berez osagaiekin datozenak onartzen ditu (adib. admin test)
- Pasahitzak berreskuratzeko metodo ahulak erabiltzen ditu, galderei erantzutea bezala

Identifikazio- eta autentifikazio-akatsak

Aplikazioa zaurgarria izan liteke baldin eta:

- Pasahitzak testu soilean gordetzen ditu, edo hash ahulekin
- Ez du autentikazio faktore-anitza erabiltzen
- URL-an saio identifikatzaileak jartzen ditu
- Ez ditu saioak baliogabetzen logout baten ondoren, edo aktibitaterik ez badago

Identifikazio- eta autentifikazio-akatsak: prebentzioa

- Ahal den heinean autentikazio multi-faktorea erabili
- Ez egin hedapenik hobetsitako kredentzialekin, batez ere admin erabiltzailearentzat
- Pasahitzen balioztatze errazak inplementatu, adibidez 10.000 pasahitz txarrenen zerrendan pasahitza bilatu
- Pasahitzen tamaina, konplexutasuna, eta balio-epea egoki kudeatu

Identifikazio- eta autentifikazio-akatsak: prebentzioa

- Ez dabiltzan login saiakerak mugatu edo atzeratu, Denial of Service (DoS) egoera bat sortu gabe. Ez dabiltzan saiakera gutztiak log-ean gorde eta administratzaileei abisatu Credential Stuffing eraso bat gertatzen bada

Identifikazio- eta autentifikazio-akatsak: prebentzioa

- Zerbitzariaren aldean, segurua den eta aplikazioarekin batera inplementatuta dagoen saio-kudeatzaile bat erabiltzea. Loginaren ondoren entropia handiko ausazko saioaren identifikatzaile bat sortu beharko luke. Saio-identifikatzaileek ez lukete URLan egon behar, modu seguruan biltegiratuta baizik. Identifikatzaileak baliogabetu egin beharko lirateke logout edo jarduera eza gertatu eta berehala

Identifikazio- eta autentifikazio-akatsak

- CWE-297: Improper Validation of Certificate with Host Mismatch
- CWE-287: Improper Authentication
- CWE-384: Session Fixation
- ...

Akatsak datuen eta softwarearen osotasunean

Software and Data Integrity Failures (A08:2021)

Akatsak datuen eta softwarearen osotasunean

Osotasunaren urraketetatik babesten ez duen kodea eta azpiegitura dagoenean ematen dira

Adibidez, aplikazio baten dependentziak konfidantzarik gabeko errepositorio edo CDN (Content Delivery Network)etatik jasoak izan badira

Akatsak datuen eta softwarearen osotasunean

Continous Integration (CI) / Continous Deployment (DC) pipeline batek kode maltzurra sartu ahal du aplikazioan

Aplikazio askok auto-eguneratze funtzioak dituzte, beharrezkoak diren osotasun testik gabe

Des-serializazio insegurua

Akatsak datuen eta softwarearen osotasunean: prebentzioa

Sinadurak edo antzeko mekanismoak erabili softwarea iturri ofizialetik datorrela eta aldatua ez dela izan balioztatzeko

NPM edo Apache Maven bezalako tresnek errepositorio fidagarriak erabiltzen dutela bermatzea. Posible bada segurtasun altuagoko errepositorioak erakunde barruan eduki (Adib. Nexus)

Akatsak datuen eta softwarearen osotasunean: prebentzioa

IC/DC pipelinek prozesu osoan kodearen osotasuna ziurtatzeko konfigurazio eta sarbide-kontrol egokiak dituztela bermatzea

Ziurtatu ez dela datu serializaturik bidaltzen sinatu gabe, edo osotasunaren balidazio metodorik gabe, des-serializatutako datuen aldaketa edo replay bat detektatzeko

Akatsak datuen eta softwarearen osotasunean

- CWE-829: Inclusion of Functionality from Untrusted Control Sphere
- CWE-494: Download of Code Without Integrity Check
- CWE-502: Deserialization of Untrusted Data
- ...

Akatsak segurtasunaren monitorizazioan

Security Logging and Monitoring Failures (A09:2021)

Akatsak segurtasunaren monitorizazioan

Ez dago monitorizazio egokirik, baldin eta:

- Gertaera auditagarriak, hala nola login-ak, huts egindako login-ak, edo balio handiko transakzioak ez dira log (erregistro) batean biltegiratzen
- Erroreek eta ohartarazpenek mezu desegokiak sortzen dituzte logetan (Edo ez dituzte sortzen)
- APien eta aplikazioen logak ez dira monitorizatzen portaera susmagarria detektatzeko
- Logak lokalean soilik gordetzen dira

Akatsak segurtasunaren monitorizazioan

Ez dago monitorizazio egokirik, baldin eta:

- Ez dago erantzun-prozesu automatikorik atalase jakin batzuetatik abiatuta
- OWASP ZAP bezalako tresnekin egindako pen-testing-ak ez ditu alertak sortzen
- Aplikazioa ez da gai denbora errealean martxan dauden erasoak detektatzeko edo ohartarazteko

Akatsak segurtasunaren monitorizazioan: prebentzioa

- Ziurtatu login-akats guztiak, sarbide-kontrola edo input-ak baliozkotzea log-ean erregistratzen direla erabiltzaile-testuinguru nahikoarekin, kontu susmagarriak identifikatu ahal izateko, eta analisi forentse bat egiteko denbora nahikoarekin biltegiratuta
- Log-ak soluzio automatizatuek erraz kontsumi ditzaketen moduan sortzen direla ziurtatzea

Akatsak segurtasunaren monitorizazioan: prebentzioa

- Balio handiko transakzioek jatorri-erregistro egokia dutela ziurtatzea, osotasun-kontrolekin, aldaketak prebenitzeko
- Erantzuteko eta berreskuratzeko plan bat ezarri eta inplementatzea, aldian behin probatuz

Akatsak segurtasunaren monitorizazioan

- CWE-117: Improper Output Neutralization for Logs
- CWE-223: Omission of Security-relevant Information
- CWE-532: Insertion of Sensitive Information into Log File
- CWE-778: Insufficient Logging