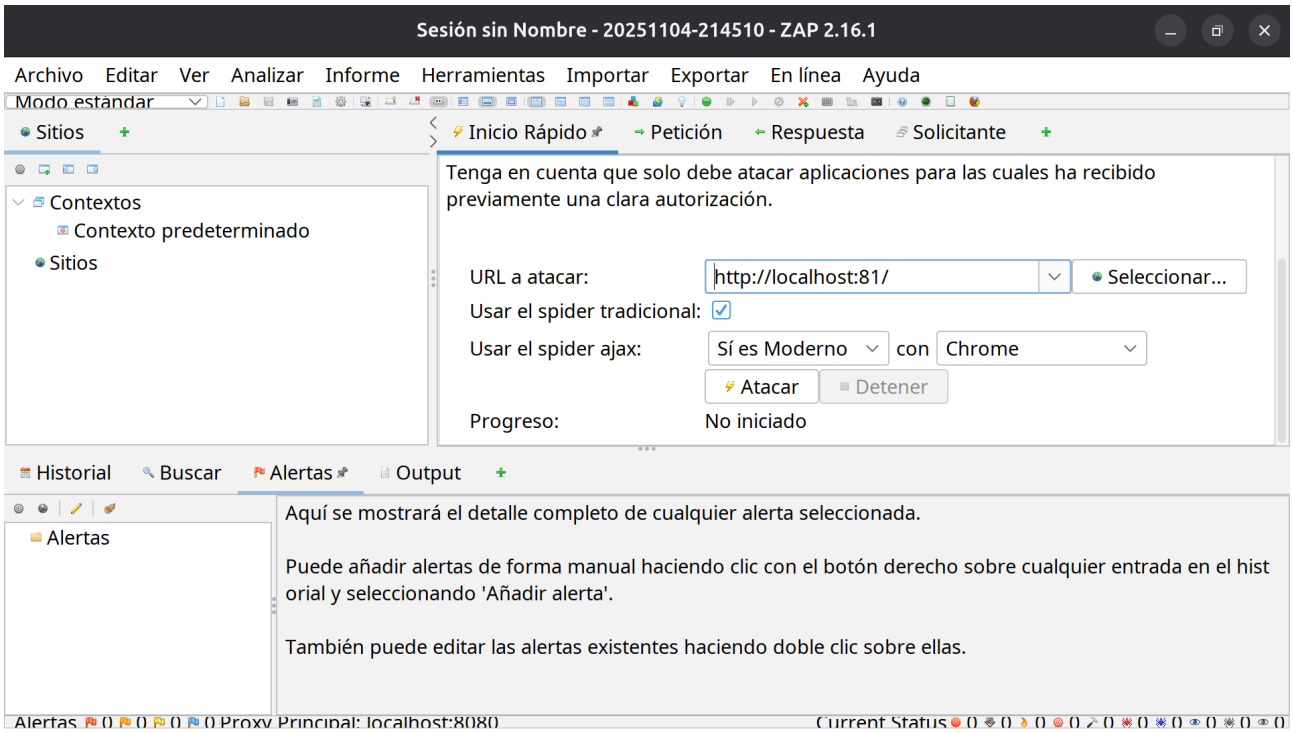
LANA 2 – WEB SISTEMAREN ANALISIA

Uriel Martin
Oihan Torrontegi

Oraingoan gure web sistemaren ahuleziak aztertuko ditugu, ZAP pentesting-tresnarekin, Open Web Application Security Project 2021 (OWASP 2021) oinarri. Lehendabizi ZAP programa deskargatu behar dugu bere web-orrialdetik. Behin instalatua, web-sistema abiarazi behar dugu, horretarako terminalean, proiektuare karpetan egonda, *docker-compose up -d* komandoa ipini.

ZAP programan, “Inicio Rapido” botoia sakatu eta web-orrialdearen linka jarri, gure kasuan http://localhost:81/ eta “Atacar” botoia sakatu.



“Alertas” erlaitzean automatikoki lortuko ditugu gure web-sistemaren ahuleziak. Gure kasuan, hauek jakianarazten dizkigu:

- Anti-CSRF Tokens-en gabezia
- Content Security Policy (CSP) goiburua ez konfiguratuta
- Anti-Clickjacking goiburuaren gabezia
- Cookie Flag HttpOnly gabe
- Cookie SameSite atributua gabe
- Zerbitzariak informazioa zabaltzen du, HTTP ""X-Powered-By"" erantzun goiburuaren bitartez
- Zerbitzariak informazioa filtratzen du “Server” erantzun goiburuaren bitartez
- X-Content-Type-Options goiburuaren gabezia
- Autentifikazio-eskaera identifikatua
- Saioa kudeatzeko erantzun identifikatua
- Web-aplikazio berritua

Orain, OWASP informea oinarri konponduko ditugu edukitako ahuleziak.

SARBIDE-KONTROLA HAUSTEA (A01:2021)

Gure webguneak oraindik ez du erabiltzaile arrunta eta administratzailea bereizteko funtzionalitatea ezarrita. Horrek sarbide-kontrol falta adierazten du, eta horrek baimendu gabeko ekintzak edo informazio mugatua eskuratzea ahalbidetu lezake. Hori saihesteko, garrantzitsua da zerbitzaritik kontrolak ezartzea, “lehenespenez ukatu” printzipioa aplikatzea, eta datu sentikorrak babestea.

AKATS KRIPTOGRAFIKOAK (A02:2021)

Ahultasun haueta kriptografiarekin edo haren gabeziarekin lotutako akatsetan oinarritzen da, informazio konfidentziala iragaztea eragin baitezakete. Ohiko arazoak dira, besteak beste, kodean pasahitz finkoak erabiltzea, zifratze-algoritmo ahulak, gakoen kudeaketa txarra, zifratu gabeko datuen transmisioa eta ziurtagirien baliozkotze okerra.

Gure kasuan, web gunea ez du ez gakoen ez datuen zifratzerik, eta horrek informazio sentikorra erakusteko arriskua dakar. Zifratze-mekanismo seguruak inplementatu behar dira, bai biltegiratutako datuetarako, bai transmititutakoetarako; horrez gain, gakoen kudeaketa egokia egin behar da, informazioa babesteko eta segurtasun-estandarrak betetzeko.

INJEKZIOA (A03:2021)

Injekzio-ahultasunak gertatzen dira aplikazio batek ez dituenean behar bezala baliozkotzen edo iragazten erabiltzailearengandik jasotzen dituen datuak, eta aukera ematen duenean komando maltzurrak datu-baseetan edo beste sistema batzuetan exekutatzeko.

Akats horiek datuak lapurtzea edo aldatzea eragin dezakete, eta sistemaren osotasunari eragin. Horiek prebenitzeko, hauek gomendatzen dira: API seguruak edo kontsulta parametrizatuak erabiltzea, datuak zerbitzarian baliozkotzea, kontsulta dinamikoetan datuak kateatzea saihestea eta segurtasun-proba automatizatuak aplikatzea aplikazioaren sarrera guztietan.

Gure web-gunean, INSERTerako prepared statement (prepare + bind_parame) erabiltzen ditugu; beraz, kontsultak ezin dira SQL injekzioaren aurrean erabili. Gainera, htmlspecialchars erabiliz balioak/erroreak erakusten ditu, eta, hala, XSS saihesten da errendatzean.

DISEINU EZ-SEGURUA (A04:2021)

Diseinu ez-seguruak esan nahi du aplikazio baten arkitekturan edo segurtasun-plangintzan akatsak daudela, gaizki diseinatutako kontrolek edo existitzen ez diren kontrolek sortuak. Inplementazio-erroreak ez bezala, arazo horiek ezin dira kode onarekin zuzendu, segurtasuna ez baitzen hasieratik kontuan hartu.

Horiei aurrea hartzeko, funtsezkoa da Secure by Design metodologiak aplikatzea, besteak beste, mehatxuen ereduak, diseinu-eredu seguruak eta eskakizunen eta baliabideen kudeaketa egokia.

SEGURTASUN-KONFIGURAZIO EZ NAHIKOA (A05:2021)

Aplikazioan segurtasun-konfigurazio eskasak, XSS erasoak, datu-injekzioak edo gaizki konfiguratutako baimenak behar ez bezala erabiltzea eragin dezake. Arrazoien artean, hauek daude: hodeiko zerbitzuen konfigurazio okerrak, beharrezkoak ez diren funtzioak edo portuak gaituta,

lehenespenez aktibatuta dauden kontuak eta pasahitzak, errore-mezu zehatzegiak eta eguneratu gabeko sistemak edo liburutegiak. Gainera, segurtasuneko HTTP goiburuak ez izateak erasoen aurreko esposizioa areagotzen du. Arrisku hauek murrizteko, ezinbestekoa da CSP goiburua ondo konfiguratzea, script-etarako, estilo-orrietarako, irudietarako eta beste baliabide batzuetarako eduki-iturri fidagarriak adieraziz.

Anti-Clickjacking goiburua falta dela detektatu da ere. Babes falta hau, erasotzaile batek aplikazioa iframe baten barruan sartzen du leku gaizto batean, erabiltzailea engainatu dezake elementu ikusezin edo manipulatuetan klik egin dezan, clickjacking izenez ezagutzen den teknika. Arriskua arintzeko, X-Frame-Options goiburua DENY edo SAMEORIGIN balioekin konfiguratu behar da, edo, bestela, CSP frame-ancestors direktiba erabiltzea edukia zein domeinutan sar daitekeen zehatzago kontrolatzeko.

Cookie Flag HttpOnly-ren gabeziak, ahalbidetzen du script maltzurrek, XSS erasoen bidez injektatutakoak adibidez, cookien edukira sartzea eta erabiltzaileen saioak lapurtzea, horrela kontuen konfidentzialtasuna eta osotasuna arriskuan jarriz. Funtsezkoa da saio-cookie guztiak HttpOnly atributuarekin markatzea, JavaScript bidez atzitu ez daitezen. Gainera, cookieak konexio seguruen (HTTPS) bidez bakarrik transmititzen direla bermatzeka gomendatzen da.

Bestalde, SameSite atributuak cookie-en bidalketa domeinu desberdinen arteko eskaeretan mugatzen du, eta nabarmen murrizten du cross-site erasoen arriskua. Hau konpontzeko, saioko cookieak SameSite=Lax edo Strict-ekin konfiguratu behar da.

Web-zerbitzariaren, reverse proxya edo tarteko geruza konfiguratzea komeni da, Server goiburuaren balioa ezkutatzeko edo aldatzeko, horren ordez balio generiko bat itzuliz (edo ezabatuz).

Gainera, X-Content-Type-Options segurtasun-goiburua falta dela detektatu da. Goiburu hori gabe, nabigatzaileek MIME sniffing egin dezakete, hau da, fitxategien eduki-mota asmatzen saia daitezke, eta, ondorioz, kodea exekutatu. Ahultasun baretzeko, beharrezkoa da X-Content-Type-Options: nosniff goiburukoa gehitzea, eta baliabide bakoitza MIME mota egokiarekin entregatzea Content-Type goiburuaren bidez.

OSAGAI KALTEBERAK ETA ZAHARKITUAK (A06:2021)

Bertsio definiturik ez duituen osagaiak erabiltzeak (adibidez, “latest” gisa konfiguratuak), aldizka ez eguneratzeak eta euskarririk gabeko softwarea erabiltzeak (sistema eragileak, web-zerbitzariak, datu-baseak, APIak, exekuzio-inguruneak edo liburutegiak) handitu egiten dute ustiapen-arriskua. Horrez gain, ahuleziak etengabe aztertzeko eta monitorizatzeko prozesurik ez egoteak edo erabilitako osagaiei buruzko segurtasun-buletinen harpidetzarik ez izateak ere eragina du erakusketan.

ZAP-ek detektatu du zerbitzariak erabilitako teknologiari buruzko informazioa zabaltzen duela HTTP “X-Powered-By” goiburuaren bidez. Informazio-mota hau erakusteak sistemaren fingerprinting-a errazten die erasotzaileei, hau da, softwarea, framework-ak edo erabiltzen ari diren bertsioak identifikatzen ditu. Hori teknologia horiekin lotutako ahultasun ezagunak aurkitzeko eta ustiatzeko aprobetxa daiteke.

IDENTIFIKAZIO- ETA AUTENTIFIKAZIO-AKATSAK (A07:2021)

Autentifikazioan, saioen kudeaketan eta CSRFen aurkako babesean faltak identifikatu dira, eta horrek handitu egiten du erasotzaile batek baimendu gabeko ekintzak egiteko arriskua, erabiltzaile legitimo baten saio aktiboa erabilita. Anti-CSRF token-ik ez egoteak gune arteko eskaerak faltsutzeko erasoak ahalbidetzen ditu, non biktimak, konturatu gabe, zerbitzarira baliozko eskaerak bidaltzen ditu. Eraso mota hau guneak erabiltzailearengan jartzen duen konfiantzan oinarritzen da eta larriagotu egin daiteke XSS ahultasunak badaude, hauek erasoa jatorri beretik exekutatzea errazten baitdituzte.

Era berean, kautotze-mekanismoetan ahuleziak daude, hala nola pasahitz ahulen edo lehenetsien erabilera, eraso automatizatuen aurkako babesik eza (indar gordina edo credential stuffing), faktore anitzeko autentifikaziorik eza eta saio edo tokenen kudeaketa desegokia, zeinak ez baitira behar bezala baliogabetzen saioa itxi edo jarduerarik ez dagoenean

Bestalde ez daukagu implementatuta rolen araberako baimenen egiaztapenik, hortaz edozein erabiltzile gure web-sisteman erregistratzean, admin-en baimenak edukiko ditu. Bereiztu behar ditugu administratzaileen eta erabiltzaile “arruntak”.

Laburbilduz, egiteko hauek ditugu:

- Token Anti-CSRFak inplementatzea
- Autentifikazioa eta saioen kudeaketa indartzea NISTen giden arabera
- Huts egindako saiakerak mugatzea