BLOG | Kim Tiedemann

EMNER

Hvordan jeg på to aftener "hackede" to store offentlige sites

Af Kim Bjørn Tiedemann 25. juni 2014 kl. 15:00

Det kræver ikke den store indsats at finde sikkerhedshuller – også på store offentlige sites, desværre. Jeg besluttede mig for at kigge nogle større sites igennem for sikkerhedshuller og de to første jeg tog fat i, havde sikkerhedshuller som kunne udnyttes af hackere. Jeg er ikke hacker, så jeg besluttede mig for at kontakte personer hos de to sites og lade dem fikse det. Derfor vil jeg også beskrive sikkerhedshullerne mere generelt uden at afsløre hvilke sites der er tale om.

Http/https mixed mode problemer

Det første site jeg kastede mig over fungerer både på http og https. Deres tanke er (tror jeg) at når man ikke er logget ind, så browser man på http sitet, mens i det øjeblik man er logget på, så skiftes til https. Et sådant site er oplagt at undersøge, for her er der flere muligheder for en angriber.

Så jeg starter med at logge ind og se om login formular hentes over en sikker forbindelse (https). En login formular skal altid loades over https og det er vigtigt at hele siden loades over https. Hvis siden loades over http, så kan en Man-In-The-Middle ændre i serverens response og dermed ændre hvor formularens indhold sendes hen eller injecte JavaScript på login siden, der sender brugernavn og adgangskode til en anden server.

I det her tilfælde loades login formularen over https og det er ikke muligt at angribe.

Efter login kommer man tilbage til sitet, denne gang på https og oppe i hjørnet på sitet kan jeg se, at jeg er logget ind. Men hvad hvis jeg ændrer protokol i adresselinjen tilbage til http? Her burde jeg ikke længere være logget ind. Desværre har det her site et sikkerhedsproblem og jeg er stadig logget ind på sitet. Det kan kun ske på en måde: Sitets authentication cookie sendes med henover den usikre http forbindelse, hvilket betyder at cookien ikke er Secure (http://en.wikipedia.org/wiki/HTTP_cookie#Secure_cookie) .

Her kan man tænke: Det er jo en aktiv handling fra brugerens side at skifte protokol. Problemet er bare at det kan ske ved at brugeren tidligere har bookmarket et usikkert link eller at jeg som hacker sender en mail med et usikkert link. Hvis brugeren allerede er logget ind (fx på en anden tab i browseren) så vil cookien blive lækket over http og dermed kan hackeren få fat i authentication cookie og bruge den over mod sitet og derfor være logget ind som brugeren.

På dette site var det dog ikke svært at få brugeren tilbage på http efter login. Flere links på sitet (specielt forsiden) pegede eksplicit på http:// og dermed ville brugeren ved at bruge sitet ganske naturligt hoppe tilbage til en usikker forbindelse.

Løsningen benytter hvad man må formode er sessionscookien (JSESSIONID er et typisk navn på en Java applikationsservers sessionscookie) som authentication cookie og det er formentligt årsagen til, at den ikke er secure. Den kan ikke være Secure, når nu sitet kører på både http og https. Sessions- og authentication cookie bør ikke være den samme cookie i det her tilfælde.

Heldigvis er cookien Http only og dermed ikke tilgængelig fra JavaScript. Dette er en rigtig god ting, da sitet jo også kunne være sårbart overfor Cross site scripting (XSS).

Hvordan kommer manden i midten?

Der findes devices som fx Pineapple Wifi (https://wifipineapple.com/) , som kan udgive sig for at være et vilkårligt access point og dermed få uvidende ofre til at forbinde deres computer eller mobile devices til access pointet.

Pineapple Wifi benytter sig af at wifi protokollen sender prober ud der spørger efter Access points, som allerede er kendt. Så har du engang forbundet til et hotels åbne wifi, så vil din computer fremover spørge om dette access point er tilgængeligt derude. Pineapple Wifi kigger på disse prober og ændrer SSID til at matche med forespørgslen. Herefter kan hackeren se alle netværkspakker der sendes mellem din computer og netværket. Det eneste der nu skal gøres er, at tage authentication cookie indholdet, vedhæfte den i requests over mod sitet og dermed være logget ind som den uvidende bruger.

Hvordan kan man sikre sig?

Det bedste råd er at benytte https konsekvent i stedet for at skifte. Dette gør sig specielt gældende hvis sitet primært er henvendt

autoriserede brugere.

Et andet råd er at authentication cookien skal være secure og også http only. Det sikrer at den aldrig sendes over en usikker forbindelse samt at den ikke kan tilgås fra JavaScript.

Et tredje råd er at benytte Http Strict Transport Security (http://en.wikipedia.org/wiki/HTTP_Strict_Transport_Security) response headeren fra serveren. Det gør at browseren herefter sikrer, at sitet kun hentes igennem en sikker forbindelse fremover (indtil tidsfristen sat i headeren udløber). Dette er dog ikke understøttet i alle browsere endnu, men det sikrer i hvertfald Chrome og Firefox brugere.

Det andet offentlige site er sårbar overfor et Cross site scripting angreb – dette indlæg må dog vente et par dage…

Om Kim Bjørn Tiedemann

Kim er udviklingschef hos Schultz og arbejder også som løsningsarkitekt og scrum coach. Han har arbejdet med agil udvikling de seneste 5 år, teknologi er hans store passion, og i sin fritid koder han på en Windows 8-app. Han er uddannet i datalogi fra AAU.

Følg @kimtiede 96 følgere