

# Hvad er?

## **Browser hijackere ...:**

Mange Microsoft Windows brugere har til stor frustration konstateret, at Internet Explorer er blevet overtaget af en såkaldt "browser hijacker". Browser hijackere overtager kontrollen og ændrer konstant startside i Internet Explorer, så den starter op på en uønsket side.

Hvis startside ændres kommer der blot nye uønskede sider, eller den samme side vender tilbage. Denne artikel kigger nærmere på fænomenet og giver anvisninger til gratis værktøjer der kan fjerne browser hijackere.

### Browser hijackere plantes via hjemmesider

Du har netop besøgt en hjemmeside, hvor du måske lidt hastigt har svaret ja til at en ActiveX komponent. Måske fordi du blev irriteret over de mange popups, eller fordi du lod dig lokke.

Den situation er der desværre alt for mange, som kan nikke genkende til. Konsekvensen er som oftest, at en lille, men potentiel skadelig applikation får tilladelse til at køre på maskinen.

Denne fremgangsmåde anvendes hyppigt af hjemmesider, som i fremtiden ønsker at sikre sig trafik og nye kunder. Det sker på din bekostning. For i fremtiden kan det tænkes, at du automatisk lander på den selv samme hjemmeside, hver gang du starter din Internet Explorer. Resultatet er til stor irritation og hvad værre er, så bliver symptomerne ofte værre og værre og er svære at slippe af med igen.

### Browser Hijackere

Denne type infektion kaldes rent teknisk for "Browsers Hijackere". På almindeligt dansk betyder det, at et lille program overtager kontrollen med din browser (ofte Internet Explorer) og guider dig mod uønskede sider.

### CWS - en ondsindet hijacker

En af de mest udbredte af sådanne programmer er CoolWebSearch (CWS). CWS kan forvandle en normal Windows maskine om til et pornografisk udstillingsvindue, idet den konstant guider brugeren mod uønskede pornografiske sider, som i sagens natur, har travlt med at åbne yderligere "popups" og reklamer, så der kan tjenes nogle hurtige penge på letpåkledte piger og drenge.

I værste fald er konsekvensen, at maskinen bliver ubrugelig, når man er på Internettet, da de sider, man i virkeligheden ønsker at kigge på, oversvømmes med pornosider og popups samt andet sjask fra udsigt til hurtige penge via online lotteri og kasino til ansøgninger om kredit kort og let

tilgængelige lån uden garantistillelse.

Uanset at denne type tilbud er uønsket for de fleste brugere og bliver hurtigt et stor irritationsmoment.

CWS browser hijackere findes i mere end 100 forskellige varianter. Det store antal gør det selvfølgelig vanskeligt at sikre sig, at et program kan fjerne alle varianter. Af samme årsag kommer populære anti-spyware programmer, som Adaware og Spybot ofte til kort over for sådanne hijackere.

CWS er en ondsindet browser hijacker familie. Hijackeren gør brug af forskellige teknikker, som kan sløre dens tilstedeværelse på maskinen og dermed gøre det vanskeligere at fjerne den igen.

En af disse teknikker går bl.a. ud på, at spærre adgang til sider, som hjælper til at fjerne "browser hijackere". På den måde kan man som bruger ikke umiddelbart få adgang til sider, som CWS ikke ønsker man skal besøge.

Derudover tilføjer den løbende uønskede sider til Internet Explorers "trusted zone/Websider du har tillid" der stort set tillader afvikling af alle former for aktivt indhold fra de hjemmesider, der oprettes i denne zone. Derved skabes der risiko for en ond spiral, der suger alskens uønskede komponenter til sig indtil PC'en stort set bliver ubrugelig.

Skjuler sig fra joblisten

Mange af de nyere og mere avancerede varianter af CWS er desuden i stand til at skjule sig fra joblisten/taskmanager.

Joblisten, eller taskmanager på engelske udgaver af Microsoft Windows, kan frembringes ved at trykke ctrl+alt+delete, eller ved at højreklikke på proceslinjen.

Den kan bruges til at danne sig et overblik over, hvilke programmer der kører i systemets hukommelse. Via joblisten kan man afslutte jobs man ikke ønsker skal køre i hukommelsen.

Det har producenterne bag CWS naturligvis fundet ud af, så de har som modsvar valgt at skjule processen i joblisten. Det sker ved at indskyde CWS programmet, som en underproces af et legitimt program - f.eks. Internet Explorer. Det kan ske via Browser Helper Objects (BHO), eller ved at køre programmet som en DLL under explorer.exe processen. De fleste brugere vil nu være tabt, så lad os i stedet kigge på, hvad man kan gøre for at slippe af med dette utøj.

AdAware og Spybot

Hvis man er plaget af symptomer, der minder om dem, der er beskrevet i denne artikel, kan man forsøge at fjerne problemer med enten AdAware eller Spybot (se links for informationer om hvor disse programmer kan hentes).

Som tidligere beskrevet, er det desværre ikke sikkert, at de er tilstrækkelige. Hvis man uden held har forsøgt at fjerne en "browser hijacker" med spybot eller Adaware kan man prøve med CWS Shredder.

#### CWS Shredder

Som navnet antyder er dette programmet skrevet med det formål at bekæmpe CWS baserede browser hijackere. Programmet er gratis og opdateres i takt med, at nye varianter af CWS fremkommer. Adressen, hvor CWS Shredder kan hentes, fremgår af links.

#### Browser renser fra CSIS

CSIS har også skrevet et gratis rens program, der kan fjerne uønskede startsider i Internet Explorer. Programmet, som er en lille .reg fil, kan køre under samtlige udgaver af Microsoft Windows. Det lille program nulstiller startsidene i IE, ligesom den ændrer forskellige værdier i registreringsdatabasen, der normalt sættes af typiske browser hijackere.

#### Dansk hjælp hos Spywarefri

Hvis man går i stå og ikke kan komme videre, så kan det anbefales at besøge <http://www.spywarefri.dk>. Denne hjemmeside drives af frivillige, der giver gode råd til at slippe af med spyware, adware eller browser hijackere. På siden kan der hentes yderligere links til gratis programmer der kan være behjælpelige med at fjerne browser hijackere.

Vy 73 de [oz1dis](#)

## SQL injektion ...:

SQL injektion er en teknik, hvor en remote eller lokal bruger, forsøger at sende SQL kommandoer igennem en webapplikation til en backend database server. En web applikation, som er sårbar overfor SQL injektion gør det muligt, at sende SQL queries og eller kommandoer til backend database serveren igennem en frontend server. SQL injection er en følgevirkning af manglende input validering. SQL står for Structured Query Language.

Avancerede server-side teknologier, som ASP.NET og kraftfulde databaser herunder MySQL og MSSQL gør det muligt for udviklere, at skabe data drevet, dynamiske webservere. Teknologien åbner mange muligheder for bruger tilpassede funktioner og kan således skræddersys til at give en mere dynamisk oplevelse. De mange muligheder skaber samtidig risiko for udnyttelse af sårbarheder i webapplikationer.

Hvis en webapplikation er sårbar over traditionel SQL injektion er det muligt for en vilkårlig bruger at sende et særligt udformet brugernavn eller password felt, som vil ændre SQL queryen og derved potentielt give os rettigheder vi ikke har. Mange Internet sider gemmer på white papers og "howtodo" dokumenter som trin for trin og i detaljeret form beskriver hvordan SQL injektion angreb kan udføres. De mange sårbare sider, som er modtagelige overfor SQL injektion er derfor lavt-hængende frugter for mange script kiddies. I fremtiden vil vi derfor se en stigning i sabotage af databaser og webside defacements som konsekvens af mangelfuld input validering.

Hvor kan SQL injektion svagheder optræde?

I princippet på alle dynamiske hjemmesider, hvor det er muligt at afsende data til en bagvedliggende server. Man kan starte med at lede efter login forms, søge funktioner, feedback osv. Disse vil typisk transmittere data til backend servere via f.eks. en method post i en form.

I et klassisk design vil man udvikle to sider, en login HTML og en ASP side. ASP siden (Login.asp) foretager et opslag i den bagvedliggende database. Et eksempel på kode, som gør som beskrevet findes herunder:

Login.htm

```
&lt;form action="Login.asp" method="post"&gt;
  Username: &lt;input type="text" name="txtUsername"&gt;&lt;br&gt;
  Password: &lt;input type="password" name="txtPassword"&gt;&lt;br&gt;
  &lt;input type="submit"&gt;
&lt;/form&gt;
```

Login.asp

```
&lt;%
  Dim p_strUsername, p_strPassword, objRS, strSQL
```

```

p_strUsername = Request.Form("txtUsername")
p_strPassword = Request.Form("txtPassword")

strSQL = "SELECT * FROM CSISUsers " & _
        "WHERE Username='" & p_strUsername & _
        "' and Password='" & p_strPassword & "'"

Set objRS = Server.CreateObject("ADODB.Recordset")
objRS.Open strSQL, "DSN=..."

If (objRS.EOF) Then
    Response.Write "Ugyldigt login."
Else
    Response.Write "Du er logget på som " & objRS("Username")
End If

Set objRS = Nothing
%&gt;

```

Selv om koden ved første øjekast ikke indeholder umiddelbare sikkerhedshuller, er det oplagt til SQL injection. Faktum er, at bruger input, som sendes via "Login.asp", direkte danner SQL statement, og som sådan åbner det mulighed for at kontrollere det statement som sendes til SQL serveren. Det kan være fatalt.

Et eksempel kan være følgende:

I felterne brugernavn/password angives: ' or '='.

SQL statementet ville derved ændre sig til:

```
SELECT * FROM CSISUsers WHERE Username=' or '=' and Password = " or "="
```

En query, som vil returnere alle records fra CSISUsers, og som samtidig vil logge brugeren ind som den første rekord i tablen.

Blind SQL injektion

I modsætning til traditionelle SQL injektion teknikker, så tager blind SQL injektion et andet udgangspunkt. Særlige SQL injektion forsøg dannes typisk på baggrund af fejlmeddelelser som kan trækkes ud af backend serveren.

I nyere tid har flere derfor praktiseret den teknik at fejlmeddelelser gemmes/undlades. En almindelig praksis som er blevet udbredt, idet de fleste detaljerede tekniske papirer om SQL injektion tager udgangspunkt i indsamling af informationer via fejlmeddelelser. Blind SQL injektion er baseret på det faktum, at det er muligt at foretage succesfuld SQL injektion på trods af, at fejlmeddelelser undertrykkes. Som navnet antyder er der tale om en teknik hvor man i princippet opererer med blind for øjnene. En tilstand som kun er muliggjort af det faktum, at SQL injektion ikke er raket-videnskab.

Blind SQL injektion dækker over det faktum, at man absolut intet kendskab har til den bagvedliggende database, herunder software og version, tabelstruktur osv.

En teknisk gennemgang af Blind SQL injektion kan findes i slutningen af denne artikel. Kig efter "Blind SQL Injection Whitepaper" fra Imperva.

#### Beskyttelse mod SQL injektion

Den mest effektive beskyttelse er at lave ordentlig input-validering på frontend serveren. Sørg for at filtrere på "enkeltstående anførselstegn", som: """" eller andre tegn, som du ikke ønsker sendes videre uden filtrering til databasen.

Vy 73 de [oz1dis](#)

## XSS ...:

Cross Site Scripting, ofte refereret til som XSS, er en teknik som potentielt kan misbruges af ondsindede personer til at afvikle kode på tværs af eksempelvis websider, via dårligt skrevet og sikrede web applikationer. I virkeligheden er Cross Site Scripting ikke et særligt dækkende navn og blev opfundet, da forståelsen for teknikken var begrænset, eller en helt anden, end de udfordringer vi oplever i dag. Cross Site Scripting har ikke nogen relation til CSS (Cascading Style Sheets), men de bliver ofte fejlagtigt forbundet.

Websider, er i dag yderst komplekse og præsenteres igennem dynamisk indhold. Dynamisk indhold opnås via webapplikationer, som i mere eller mindre grad er sikre.

Det er vigtigt at forstå, at XSS ikke direkte kan lede til kompromittering af webservere. Derimod kan traditionelle XSS angreb lede til injektion af scripts, herunder JavaScript, VBScript, ActiveX, HTML eller Flash, som alle afvikles hos remote brugere med de samme rettigheder og sikkerhedsrestriktioner som brugeren har overfor den side, hvorfra scriptet leveres. En webside, der er sårbar overfor XSS, og som en bruger normalt har tillid til, kan således misbruges som medie for ondsindede scripts.

### Klassisk XSS

Et klassisk eksempel på XSS er at sende særlige parametre til et usikkert CGI script der befinder sig på en sårbar webside. CGI-scripts er små applikationer, som udfører forskellige handlinger på websiden, men som også kan misbruges til at præsenteres data på siden som ikke skulle vises eller præsenteres, ligesom de kan anvendes til at omdirigere trafik. CGI-redirectorer kan anvendes med det formål, at sende en bruger fra en webside til en anden, men kan også misbruges i forbindelse med Phishing. CGI-scripts, er som tidligere nævnt små programmer, som kører med rettigheder på webserveren, og kan derfor selvstændigt indeholder sårbarheder der kan lede til afvikling af kode med kompromittering til følge. Sidstnævnte vedrører dog sårbarheder i cgi-scriptet og er ikke relateret til XSS.

Den mest populære form for XSS opstår ved at injektive scripts i en form, som ikke forsvarligt validerer bruger suppleret data. En ondsindet bruger kan indskyde script som kan lede en bruger videre til en anden hjemmeside eller køre direkte via browseren.

Grafisk illustration:

Et par eksempler på XSS

Eksempel 1:

1. BrugerA har tillid til en-dansk-netbank.dk og tillader derfor kørsel af JavaScript fra det domæne
2. BrugerB finder en svaghed, hvorved det er muligt at injektive sit eget javascript på domænet [www.en-dansk-netbank.dk](http://www.en-dansk-netbank.dk). Javascriptet kan anmode brugeren om pinkode og kontooplysninger til eksempelvis "En dansk netbank.dk". Da script præsenteres fra en-dansk-netbank.dk kan denne form for

social engineering have en yderst stor effekt og mange brugere vil hoppe på og indtaste oplysninger. De indtastede oplysninger, som lander i det script som brugerB har injektet, videresender data til et andet domæne, hvorfra oplysningerne kan høstes.

På let vis, har BrugerB altså opnået adgang til følsomme oplysninger, som brugerA mener at have givet til [www.en-dansk-netbank.dk](http://www.en-dansk-netbank.dk)

Eksempel 2:

1. BrugerA har en konto hos en-dansk-netbank.dk og er logget ind
2. BrugerB injekter et JavaScript for at indsamle session ID og sende det til brugeren
3. BrugerA besøger den manipulerede side hvorfra session ID sendes til BrugerB
4. BrugerB kan nu anvende BrugerA's konto indtil brugerA logger ud eller skifter password

XSS kan altså misbruges til brugerkonto hijacking, manipulation af brugerindstillinger og præferencer, cookie tyveri/poisoning eller reklame præsentation. På det seneste er der leveret forskellige proof-of-concept kode der dokumenterer at XSS også kan anvendes til Denial of Service.

Et par eksempler på XSS, hvor vilkårlig javascript kan injektes, kan findes herunder (NB der er tale om fiktive URLs. CSIS.dk er naturligvis ikke sårbar overfor disse PoCs).

[http://www.csis.dk/modules.php?](http://www.csis.dk/modules.php?name=Downloads&d_op=viewdownload&lid=02&ttitle=[JAVASCRIPT])

[name=Downloads&d\\_op=viewdownload&lid=02&ttitle=\[JAVASCRIPT\]](http://www.csis.dk/modules.php?name=Downloads&d_op=viewdownload&lid=02&ttitle=[JAVASCRIPT])

[http://www.csis.dk/modules.php?](http://www.csis.dk/modules.php?name=Downloads&d_op=ratedownload&lid=118&ttitle=[JAVASCRIPT])

[name=Downloads&d\\_op=ratedownload&lid=118&ttitle=\[JAVASCRIPT\]](http://www.csis.dk/modules.php?name=Downloads&d_op=ratedownload&lid=118&ttitle=[JAVASCRIPT])

[http://www.csis.dk/modules.php?](http://www.csis.dk/modules.php?op=modload&name=Members_List&file=index&letter=[JAVASCRIPT])

[op=modload&name=Members\\_List&file=index&letter=\[JAVASCRIPT\]](http://www.csis.dk/modules.php?op=modload&name=Members_List&file=index&letter=[JAVASCRIPT])

[http://www.csis.dk/submit.php?subject=](http://www.csis.dk/submit.php?subject=[JAVASCRIPT]&story=[JAVASCRIPT]&storyext=[JAVASCRIPT]&op=Preview)

[\[JAVASCRIPT\]&story=\[JAVASCRIPT\]&storyext=\[JAVASCRIPT\]&op=Preview](http://www.csis.dk/submit.php?subject=[JAVASCRIPT]&story=[JAVASCRIPT]&storyext=[JAVASCRIPT]&op=Preview)

[http://www.csis.dk/user.php?op=userinfo&uname=](http://www.csis.dk/user.php?op=userinfo&uname=[JAVASCRIPT])

[\[JAVASCRIPT\]](http://www.csis.dk/user.php?op=userinfo&uname=[JAVASCRIPT])

## XSS cookie tyveri

Som nævnt tidligere er det via XSS også muligt at gennemføre cookie tyveri. Denne teknik kan være særligt farlig, idet flere webapplikationer efter autentifikation, gør brug af cookies. Hvis en webside er sårbar overfor XSS scripting og samtidig gør brug af cookies, er det muligt at stjæle disse og udføre handlinger, som den bruger der er logget ind på systemet.

Cookies er begrænset til domæner og websider. XSS svagheder på et domæne kan give adgang til cookie data:

JavaScript Expression: "document.cookie"

window.open

document.img.src

Hidden Form Submit

[http://www.csis.dk/cgi-bin/cookie\\_thief.pl?COOKIEACC](http://www.csis.dk/cgi-bin/cookie_thief.pl?COOKIEACC)

Cookie data sendes via et GET request til et CGI script til en modtager udenfor domænet/websiden.

Hvad er forskellen på XSS og script injektion?

Som hovedregel gælder det, at script injektion introducerer en permanent tilstand, hvor det er muligt direkte at modificere/ændre indhold på en sårbar hjemmeside. I direkte modsætning vil misbrug af en XSS svaghed være temporær. Der åbnes altså ikke mulighed for direkte at ændre indhold på en sårbar webside.

Hvad sker der hvis jeg ikke lukker et XSS hul på min webside?

Hvis eksempelvis Kruse Online/Onsite Scanner har fundet en XSS sårbarhed på din webside, så bør man lukke hullet snarest muligt. Hvis man ikke vælger at følge de anvisninger som gives i relation til sikker webapplikationsprogrammering, så er websiden i risiko for at brugere der anvender tjenesten, får deres data lækket, ændret eller slettet, ligesom - afhængig af sårbarhedens omfang - det er muligt at misbruge XSS svagheden til cookie tyveri. Det gør det muligt at udføre samme handlinger på webserveren som den bruger cookien er udstedt til. En XSS svaghed på din webside, kan ligeledes blive publiceret på sikkerhedsrelaterede nyhedslistes, og derved skade firmaets ry og omdømme.

Praktiske eksempler på XSS

Hos Point Blank Security har man samlet en række eksempler på XSS, som forefindes på store kommercielle websites. Listen med eksempler kan ses på følgende URL: <http://www.pointblanksecurity.com/xss/>

En helt ny og opdateret liste er netop blevet frigivet og kan findes på adressen:

<http://pointblanksecurity.com/xss/xss2.php>

Vy 73 de [oz1dis](#)

# Dette er en underside til:

<http://www.oz1dis.dk/>

Allan I Jensen  
Vinterbuen 5  
2750 Ballerup  
Telf 44654488  
Mob 40684488  
Skype: OZ1DIS  
Hamcall sign: OZ1DIS

