PERSONVERN, GDPR OG COREPUBLISH

Innledning

Dette dokumentet beskriver hvordan CorePublish fungerer i forhold til personvernloven og GDPR.

Det er viktig å understreke at CorePublish er et CMS og et rammeverk for å bygge nettsteder. Det kan derfor være gjort tilpasninger og utvidelser rundt funksjonalitet til hvert enkelt nettsted som ikke er beskrevet her. Dette må eiere at de respektive nettstedene kontrollere og evt. dokumentere.

Personvern

Bruk av cookies på nettsteder laget med CorePublish

- For å få nettløsningen til å fungere settes en cookie som vanligvis navngis ***CorePublishSession.*** **Dette er en «sesjonscookie» som** fjernes fra datamaskinen når brukeren lukker nettleseren. Denne cookien har en referanse-id til en sesjonsfil på webserveren. Sesjonsfilen vil for anonyme brukere ikke inneholde personopplysninger, men kan på enkelte nettløsninger bli brukt til å mellomlagre eksempelvis skjemadata, søketekst, ip-adresse og user agent. Sesjonsfilen slettes automatisk senest 15 minutter etter at brukeren har lukket nettleseren eller vært inaktiv, og lagres således ikke permanent.

- For å identifisere hvilken type enhet og nettleser besøkende har kan cookien ***ctdevicecachekey*** settes. Denne cookien har en varighet på en uke og gjør det mulig for løsningen å presentere innhold til riktig type enhet (pc, mobil, nettbrett, etc.). Denne cookien består kun av data nettleseren allerede har sendt fra seg (user agent), og den lagres kun på klientens nettleser og ikke på serveren.

Data om innlogging

Alle påloggінger, både i CorePublish (behandlingsansvarlige sine brukere av CorePublish) og på nettstedene (behandlingsansvarlige sine brukere på nettstedet, hvis slik funksjonalitet benyttes), logges i en egen databasetabell. Hensikten er både å kunne oppdage mislykkede påloggingsforsøk, oppdage og stoppe forsøk på innbrudd og å kunne se når og om en bruker sist logget seg på. Eksempelvis kan man se om en bruker er på riktig url, og om han skriver riktig brukernavn ved pålogging. Dataene brukes også for å kunne se hvem som logget seg inn i løsningen på et gitt tidspunkt.

Påloggingstabellen lagrer følgende data:

- Brukernavn
- Dato og tidspunkt for pålogging
- Hvilken url som ble benyttet ved pålogging.
- Brukerens ip-adresse (hvis aktivert, som standard avslått)
- Brukerens user agent (nettleser) (hvis aktivert, som standard avslått)

Disse dataene lagres inntil behandlingsansvarlig ber om å få dem slettet.

Statistikk-modulen
Dersom behandlingsansvarlig benytter skjemamodulen i CorePublish, vil denne lagre en rekke data om brukeren. Statistikkmodulen samler først en mengde detaljerte data i en rådatatabell i databasen.

Rådataene lagrer følgende data som kan kvalifisere som persondata for hver sidevisning:

- Hvilken url/nettside som ble besøkt
- Dato og tidspunkt for sidevisning
- Brukernavn (hvis brukeren er pålogget)
- Referer (hvilken side brukeren kom fra) og eventuelle søkeord hvis han kommer fra en søkemotor
- Sesjonsid
- Brukerens ip-adresse (hvis aktivert i modulen, som standard avslått)
- Brukerens user agent (nettleser) og detaljer om nettleseren (hvis aktivert i modulen, som standard avslått)
- Hvilke objekter i CorePublish (artikler, kategorier, bilder) som ble vist på den aktuelle siden

Disse dataene blir så aggregert til ulike oppsummeringstabeller, og rådataene blir så slettet etter 2 dager.

Summeringstabellene som blir aggregert inneholder svært lite persondata, og per nå benytter ikke summeringstabellene all data fra rådata-tabellene. Eksempelvis user agent blir aldri aggregert med mindre kunden selv har laget egne rapporter på dette. Dersom valget «Hvilke nettverk besøk kommer fra» er aktivert, vil det bli lagret hvilke ip-adresser som har besøkt hvilke artikler, kategorier og multimedia-filer. Summeringstabellene blir liggende permanent i løsningen.

Skjemaer
Dersom behandlingsansvarlig benytter skjema-modulen i CorePublish, vil innsendte skjema kunne inneholde persondata.

Disse skjemaene og tilhørende felter kan redigeres av behandlingsansvarlig og det er således behandlingsansvarlig som har det fulle ansvaret for å vurdere personvernhensyn rundt hvilke data de ulike skjemaene i løsningen samler.

Alle innsendte skjemaer og tilhørende data vil som standard bli lagret i en database-tabell.

I tillegg til de feltene som er definert unikt per skjema (eksempelvis «navn» og «adresse») så vil følgende data alltid bli lagret om hvert skjema:

- Hvilken url/nettside som ble besøkt
- Dato og tidspunkt når skjemaet ble sendt inn
- Brukernavn og e-post (hvis brukeren er pålogget)
- Brukerens ip-adresse (hvis aktivert i modulen, som standard avslått)
- Brukerens user agent (nettleser) (hvis aktivert i modulen, som standard avslått)

Disse dataene blir liggende inntil behandlingsansvarlig velger å slette dem fra CorePublish, eller velger å bruke den innebygde funksjonaliteten for å slette skjemaer (enten med en gang eller etter x antall dager).

I tillegg kan innsendte skjemaer bli sendt på e-post og/eller konfigureres til å lagres med andre lagringsmetoder dersom behandlingsansvarlig konfigurerer dette. Det er i så henseende behandlingsansvarlig sin plikt å sørge for at disse lagringsmetodene, eksempelvis utsending, mottak og behandling av e-poster, skjer forskriftsmessig.

Sikkerhet i CorePublish

Generelt

Under utviklingen av CorePublish har sikkerhet alltid hatt en høy prioritet. CoreTrek som eier av applikasjonen er et svært sikkerhetsbevisst selskap med stor kompetanse på området som dreier seg om sikkerhet på web.

Validering av brukerinput

Ett av de viktigste elementene når man skal utvikle en sikker løsning, er å validere dataene som kommer fra brukeren. Mange sikkerhetshull er forårsaket av dårlig validering av de data som brukeren sender til applikasjonen, og åpner dermed for ”hull” i løsningen. CoreTrek har derfor lagt ned mye arbeid i å sørge for en sikker validering av data som blir sendt til applikasjonen. Alle parametre sendes gjennom en kontroll som verifiserer at paremeterets type(tall/streng), verdi (lengde, gyldige tegn) og sendingsmåte er korrekt. Dersom et av parametrene ikke tilfredstiller de kravene som er satt i kontrollen, blir programkjøringen avbrutt og brukeren blir gitt en feilmelding. Denne sikkerhetssjekken vil som en konsekvens også sørge for å øke kvaliteten på de dataene som sendes fra brukeren, da innholdet av det brukeren sender til en viss grad vil være sikret riktig format (f.eks. at et postnummer kun består av tall). Koden som håndterer dette er sentralisert slik at eventuelle feilrettinger og forbedringer i valideringen kommer hele applikasjonen til gode.

Brukere og passord

Alle som skal bruke CorePublish, må logge seg på applikasjonen med domene, brukernavn og passord. Det må settes opp en brukerkonto for hver bruker. For hver brukerkonto må det angis et brukernavn og et passord. Brukernavnet må være unikt. Passordet må være minst 5 tegn, dette for å unngå at brukere skriver inn ”lette” passord som senker sikkerheten. CorePublish har også en «password policy» funksjon som sjekker at passordet er godt nok, her sjekkes at passordet ikke er likt brukernavnet, at det inneholder både tall og bokstaver osv. Denne funksjonaliteten er pluggbar og kan byttes ut til egne regler for de kundene som ønsker dette.

Alle passord i CorePublish lagres med en såkalt MD5-kryptering. Dette er en sterk kryptering som i praksis ikke er reversibel, det vil si at man klarer ikke å finne det opprinnelige passordet ut fra den lagrende, krypterte strengen. Dette vil si at selv om uvedkommende skulle klare å få tilgang til brukerdatabasen så vil man ikke kunne lese ut brukernes passord.

Innlogging

Innloggingen er en kritisk del av enhver applikasjon sin sikkerhet, og det er derfor lagt ned mye arbeid i å gjøre innloggingen i CorePublish så sikker som mulig. Både domene, brukernavn og passord valideres etter strenge regler og kun godkjente tegn som a-z, 0-9 og enkelte andre tegn (., @, - og _) godtas. Brukeren vil få en feilmelding dersom ugyldige tegn blir forsøkt sendt til CorePublish. Når dataene fra brukeren er validert, utføres selve autentiseringen mot brukerdatabasen (eller andre valgte kilder som f.eks. LDAP). Dersom brukeren har oppgitt gyldige data, blir en sesjon startet og brukeren blir logget inn og sendt til forsiden i CorePublish.

CorePublish har også støtte for å aktivere såkalt CAPTCHA (http://en.wikipedia.org/wiki/CAPTCHA). Dette gjør at brukeren må tolke et bilde med tekst og skrive dette inn som en del av

brukerauthentiseringen. Dette vanskeliggjør maskinbaserte angrep, fordi man da i tillegg må lese og og tolke teksten på bildet.

CorePublish krever automatisk CAPTCHA ved innlogging fra ip-adresser som har mer enn 3 mislykkede innloggingsforsøk sammenhengende.

To-faktor autentisering
CorePublish støtter også to-faktor autentiseringen (TFA) hvor brukeren får tilsendt en engangskode på SMS eller e-post. Ved at brukeren må skrive inn en unik engangskode så økes sikkerheten betraktelig. Tofaktor autentiseringen er laget etter TOTP-standarden.

Begrensning på land (geo-ip)
CorePublish støtter å begrense hvilke land en brukergruppe får logge inn fra. Dette gjør at man kan stenge ute enkelte land – eller bare tillate innlogging fra gitte land for å ytterligere øke sikkerheten.

Kun en sesjon per bruker
CorePublish kan begrense antall sesjoner per brukernavn. Dette gjør at ett brukernavn kun kan være i bruk fra en enhet(PC, mobil etc) om gangen. Dette vil f.eks. gjøre det vanskeligere at en brukerkonto blir (mis)brukt av flere personer samtidig.

Sesjonshåndtering - når brukeren er innlogget
Når brukeren er innlogget i CorePublish, får han tildelt en unik sesjon mot webserveren. Hver sesjon har en ID, og denne blir lagret i en cookie på brukerens PC. Ved navigering i applikasjonen er det denne sesjons-ID'en som forteller webserveren at han snakker med en gyldig bruker. Dersom webserveren får en forespørsel fra en bruker som ikke oppgir en gyldig sesjons-ID, blir sesjonen avsluttet og brukeren blir omdirigert til login-skjermbildet. Denne gyldighets-sjekken på sesjonen er det første som skjer uansett hvor brukeren befinner seg i applikasjonen, og dette sikrer at ugyldige brukere ikke har mulighet til å aksessere applikasjonen eller sende parametre til den. Dersom en bruker ikke fornyer sin sesjon innen ca 30 minutter (konfigurbart pr installasjon), blir sesjons-ID'en gjort ugyldig og brukeren vil bli oppfattet som en ikke-innlogget bruker.

Særlig beskyttelse mot kjente "hacker-metoder"
Nedenfor følger en beskrivelse av metoder som er mye brukt for å skade eller ta kontroll over webapplikasjoner, og hvordan CorePublish er beskyttet mot disse.

1 Beskyttelse mot "SQL injection"
CorePublish sin login-rutine er beskyttet mot såkalt "SQL injection" (http://en.wikipedia.org/wiki/SQL_injection). Dette er en metode som blir brukt av "hackere" for å ta kontroll over eller skade et fremmed system. Metoden går kort fortalt ut på å sende spesielle tegn/data til applikasjonen ved f.eks. innlogging. Når dette blir videresendt til databasen, så vil det av databasen oppfattes som en annen kommando enn det som opprinnelig var meningen, og man kan utnytte dette til f.eks. å slette eller forandre data.

Ved at CorePublish kun godtar godkjente og sikre tegn i det som blir sendt fra brukeren, vil dette aldri kunne oppstå ved innlogging i CorePublish.

Videre er også alle andre spørringer som utføres, både fra inni CorePublish og fra presentasjonslaget, sikret mot Sql injection. CorePublish har sitt eget API(funksjonssett) for SQL-spørringer kalt CtSQL, som benyttes for alle spørringer som utføres. Dette har en innebygget støtte for å escape input slik at SQL injection unngås. Det kjøres nattlige tester hver natt som tester denne funksjonaliteten.

2 Beskyttelse mot "Cross Site Scripting"

"Cross Site Scripting" (http://en.wikipedia.org/wiki/Cross-site_scripting) baserer seg på at applikasjonen av og til har behov for å vise informasjonen den mottar som et parameter, et sted på den HTML-siden som blir generert. F.eks. dersom innloggingen feiler pga feil brukernavn eller passord, så blir man sendt tilbake til login-siden med brukernavnet som et parameter, slik at dette vises i innloggingsskjermbildet og brukeren slipper å skrive dette på nytt. Ved å sende script-kommandoer i dette parameteret, vil en hacker kunne klare å få startet et script på siden som ser ut som det kommer fra applikasjonen selv. Siden det ser ut som det kommer fra applikasjonen, vil brukere ikke "reagere" på det som skjer fordi man antar at man er på en sikker side. I virkeligheten kan f.eks. scriptet snappe opp og videresende brukerens passordet. I CorePublish er man sikret mot dette ved at all bruker-input blir rensket før den blir vist til brukeren igjen. Dette sikrer at f.eks. Script-tagger osv blir ufarliggjort.

CorePublish blir også periodisk testet med en tilleggsmodul til PHP som kjører i «tainted mode», og som effektivt avslører eventuelle svakheter i forbindelse med brukerinput som ikke blir håndtert riktig.

3 Beskyttelse mot "Man in the middle-angrep"

Et såkalt "man in the middle attack" utføres ved at en tredje part (person eller maskin) klarer å lytte på ukryptert trafikk mellom brukeren og webserveren, og klarer å fange opp sesjons-ID'en som sendes fra brukeren til serveren. Personen vil så selv sende denne sesjons-ID'en til serveren med det formål å utgi seg som den faktiske brukeren, og få tilgang til det som den opprinnelige brukeren hadde tilgang til. I CorePublish er man sikret mot dette ved at det på serveren lagres tilleggsdata for hver sesjons-ID som sjekkes for hver gang den blir sendt fra brukeren. For eksempel dersom CorePublish plutselig mottar den samme sesjons-ID'en fra en annen adresse enn den kom fra forrige gang, så tas dette som et tegn på et slikt angrep. Sesjons-ID'en blir da umiddelbart ugyldig, og brukeren blir omdirigert til login-skjermbildet. CorePublish kan også selvsagt kjøres via https/SSL, noe som vanskeliggjør denne typen angrep fordi all kommunikasjon mellom bruker og webserver skjer kryptert.

4 Beskyttelse mot Cross Site Request Forgery (CSRF)

Såkalte Cross Site Request Forgery-angrep (http://en.wikipedia.org/wiki/Cross-site_request_forgery) utføres ved at andre nettsider utnytter at brukeren er logget inn i en annen applikasjon. For eksempel kan en ondsinnet side prøve å gjøre forespørsler mot en nettbank, og hvis brukeren da tilfeldigvis er logget inn på sin nettbank så kan det bli gjort handlinger uten at brukeren engang ser at de blir utført (les detaljert forklaring på Wikipedia).

CorePublish er sikret mot CSRF ved at hver handling krever en unik CSRF-nøkkel som blir generert på den foregående siden (den som leder til handlingen). Dermed klarer ikke andre nettsider å utføre handlinger mot CorePublish fordi de har ikke tilgang til å generere og sende den rette CSRF-nøkkelen.

Informasjonssikkerhet i CorePublish

CorePublish har innebygde rutiner som sørger for at innloggede brukere ikke klarer å få tilgang til deler av systemet de ikke har tilgang til. Sjekker for modultilgang og riktige rettigheter ligger bygget inn i applikasjonsrammeverket og sørger for at brukere ikke klarer å få tilgang til andre deler av systemet hvor de ikke skal ha tilgang.

Dette gjelder både i selve CorePublish-applikasjonen og på frontend (nettstedet). Innebygget tilgangstyring gjør at informasjon ikke kan endres av brukere som ikke har tilgang – og at kun brukere med riktig lesetilgang får tilgang til å lese informasjonen.

Rutinene som går på sikkerhet ligger bygget inn dypt i CorePublish sitt programmerings-API. Dette for å sikre at utviklere som programmerer på løsningen heller ikke skal klare å hente ut og vise feil

i

n

f

o

r

m

a

s

j

o

n

v

e

d

e

t

u

h

e

l