OS2faktor MFA

Exit Strategi

**Version**: 1.1.0

**Date**: 23.04.2022

**Author**: BSG

# Formål

Dette dokument har til formål at beskrive de aktiviteter der er nødvendige for at overdrage driften af OS2faktor MFA infrastrukturen til en anden driftsleverandør end den nuværende.

Dokument forholder sig ikke til kontraktuelle forhold, men alene til aktiviteter, processer og opgaver som skal udføres i forbindelse med overdragelsen af driften fra én driftsleverandør til en anden.

# Omfang

Dette dokument omfatter alene OS2faktor MFA løsningen, og dermed ikke andre omliggende komponenter, som OS2faktor Login, OS2nykode eller 3.part integrationer mod OS2faktor MFA udarbejdet af leverandører udenfor OS2 samarbejdet.

I praksis er følgende komponenter indbefattet af dokumentet

* OS2faktor MFA klienterne
  + Chrome browser udvidelsen
  + Native-windows klienten
  + iOS klienten
  + Android klienten
* OS2faktor backenden
  + Backend API komponenten
  + Selvbetjeningsportalen
  + Administratorportalen
  + Yubikey integrationen
* Connectors
  + AD FS connectoren
  + RADIUS connectoren
* os2faktor.dk domænet

Kildekode til OS2faktor MFA løsningen forefindes på Github her

<https://github.com/OS2faktor/OS2faktor-MFA>

# Overdragelse af OS2faktor MFA klienterne

Der findes 5 forskellige MFA klienter, fordelt på 4 forskellige kodebaser, hvor én af disse (Yubikey klienten) er indarbejdet i backenden og derfor ikke beskrevet i dette afsnit.

## Chrome browser udvidelsen

Chrome browser udvidelsen er lavet i Javascript, og kildekoden til denne ligger i Github repositoriet, i denne underfolder

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/clients/chrome-extension>

I koderepositoriet findes et shellscript (build.sh) til at bygge et release baseret på den aktuelle kildekode. Bemærk at et release af en Chrome udvidelse blot er en zip-fil, indeholdende kildekoden i en bestemt struktur. Shellscriptet sikrer blot strukturen og danner zip-filen. Man kan nemt foretage den samme opgave via manuelle trin, da det blot består af at kopiere 3 filer/foldere og så zippe disse.

Man skal redigere manifest.json filen i forbindelse med et nyt release, og øge versionsnummeret heri. Dernæst kan man foretage et deploye i Chrome Web Store som beskrevet i README.md filen i koderepositoriet.

**OBS!** På nuværende tidspunkt er OS2faktor MFA klienten til Chrome opretter som en udvidelse udgivet af den nuværende driftsleverandør. Man bør overveje at flytte ejerskabet af denne extension til OS2, som I så fald skal oprette sig som leverandør/udbyder i Chrome Web Store, og give driftsleverandøren adgang til at administrere denne.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå og bygge OS2faktor MFA klienten til Chrome
* De kan logge ind i Chrome Web Store
* De har adgang til at administrere OS2faktor MFA klienten inde i Chrome Web Store og hermed frigive en ny version

## Native Windows Klient

Windows klienten er lavet i C#, og kildekoden til denne findes i Github repositoriet i følgende underfolder

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/clients/win10>

Kildekoden kan åbnes i Visual Studio, hvor man kan kompilere klienten som en standard C# applikationen.

For at distribuere klienten skal man pakketere enten en EXE eller MSI installer, som kommunerne kan installere i deres windows miljøer. Der findes en lang række værktøjer til dette, og det anbefales at den nye driftsleverandør vælger det værktøj som de finder bedst egnet til formålet.

Den nuværende driftsleverandør har avendt både InnoSetup og AdvancedInstaller, hvor AdvancedInstaller er det værktøj der anvendes på nuværende setup, da der dannes MSI pakker.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå og bygge OS2faktor MFA klienten til Windows
* De kan bygge en MSI eller EXE installer af den kompilerede kode

## iOS klient

iOS klienten er lavet i Cordova, der er et Javascript værktøj til at lave cross-platform apps. Det er den samme kildekode der anvendes til både iOS og Android, men bygge- og deployment processen er forskellig.

Kildekoden findes i Github repositoriet i følgende underfolder

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/clients/os2faktor-mobile-client>

For at kompilere koden til iOS skal man bruge en Mac og XCode. Først skal man dog installere en række javascript byggeværktøjer via konsollen på sin Mac

Følgende værktøjer skal installeres

* npm (node package manager)
* cordova (installeres via npm)

I den folder man har hentet koden ud i, installerer man nu iOS platform i cordova ved at skrive

$ cordova platform add [ios@5.1.1](mailto:ios@5.1.1)

Herefter kan man åbne projektet i Xcode og kompilere projektet. Når man skal signere projektet skal man bruge et signerings og distributionscertifikat fra Apple. Den efterfølgende IPA pakke der dannes af XCode kan man deploye i Apples app store via deres normale processer.

**OBS!** OS2faktor MFA klienten til iOS er distribueret af den nuværende driftsleverandør, og man bør overveje at overdrage ejerskabet af denne distribution til OS2, og give driftsleverandøren adgang til at administrere deployment af nye versioner af app’en.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden og bygge en signeret IPA fil
* De kan tilgå Apples app store og administrere OS2faktor MFA klienten (og dermed foretage nye releases)

## Android klient

Android klienten er lavet i Cordova, der er et Javascript værktøj til at lave cross-platform apps. Det er den samme kildekode der anvendes til både iOS og Android, men bygge- og deployment processen er forskellig.

Kildekoden findes i Github repositoriet i følgende underfolder

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/clients/os2faktor-mobile-client>

Projektet kan kompileres på forskellige måder, herunder via Android Studio eller lignende Android værktøjer. Man kan dog også kompilere projektet manuelt fra kommandolinjen på følgende måde

* Installer Java 8
* Hent Android SDK commandline tools (version 6609375 er anvendt med succes til dette formål, men nyere versioner vil formodentligt også fungere)
* Installer sdkmanager build-tools og platform-tools I version 29 (29.0.2 er anvendt med succes) vha commandline tools
* Kør ”npm install” i roden af kodebasen for at installere alle afhængigheder (herunder Cordova)
* Kør ”cordova platform add [android@9.0.0](mailto:android@9.0.0)” for at installere Android platformen i Cordova
* Kør ”cordova prepare android” for at klaregøre kompileringen
* Kør ”cordova requirements” for at sikre at alt fungerer som det skal
* Kør ”cordova build –release android” for at foretage et egentlig byg af kodebasen
* Signer APK filen vha zipalign og apksigner værktøjerne. Dette forudsætter at man har et signeringscertifikat til at signere filen med

Den afsluttende APK fil er klar til at blive releaset i Google Play appstoren.

**OBS!** OS2faktor MFA klienten til Android er distribueret af den nuværende drif driftsleverandør, og man bør overveje at overdrage ejerskabet af denne distribution til OS2, og give driftsleverandøren adgang til at administrere deployment af nye versioner af app’en.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden og bygge en signeret APK fil
* De kan tilgå Google Play app store og administrere OS2faktor MFA klienten (og dermed foretage nye releases)

# Overdragelse af OS2faktor MFA backenden

Kildekoden til OS2faktor MFA backenden befinder sig i Github repositoriet her

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/backend>

Backenden består af 3 selvstændige komponenter, og 2 fællesmoduler (common og yubikey). Disse komponenter er alle skrevet i Java, og kræver Java 8 (men bliver omlagt til Java 11 ved næste release).

Man bør starte med at kompilere alle modulerne fra roden af backend kildekoden, for at sikre at alle afhængigheder (de 2 fællesmoduler) er kompileret. Til dette formål anvendes Maven, og man kan kompilere ved at skrive

$ mvn clean install

Bemærk at applikationen forudsætter eksistensen af MySQL (eller MariaDB) databaseserver, og at der er oprettet et database skema samt et tilhørende brugernavn/kodeord som applikationen kan anvende.

Konfiguration af det forventede skema, brugernavn og kodeord kan opsættes i config/application.properties filerne i de enkelte undermoduler.

## Server API

Server API’et ligger i underfolderen ”backend”. Når alle modulerne er kompileret, er Server API’et det også, og man kan starte det op fra kommandolinjen ved at skrive

$ mvn spring-boot:run

Dette afvikler kodebasen “in-place”, hvilket er fint til afprøvning af kildekoden.

Når man kompilerer projektet, dannes en selvindeholdt JAR fil under ”target” folderen, som indeholder den fulde applikation, eksklusiv konfiguration. Man kan kopiere denne applikation over på en server, og så placere en konfigurationsfil i en konfigurationsfolder ved siden af, fx på denne måde

.

├── config

│   └── application.properties

└── os2faktor-backend.jar

Herefter kan man starte applikationen op ved at skrive

$ java -Dloader.path=”config/” -jar os2faktor-backend.jar

Man kan tage udgangspunkt i den konfigurationsfil der ligger i kildekodens ”config” folder, og tilpasse til ens konkrete deployment behov. Bemærk her at konfigurationsfilen peger på nogle AWS notifikationsendpoints, da OS2faktor MFA gør brug af en push-gateway i AWS miljøet til at afsende push notifikationer til Chrome, iOS og Android MFA klienterne.

Det er ikke nødvendigt at anvende AWS til at sende push notifikationer, men hvis man ikke gør dette, så skal man udvikle en direkte integration til Apple og Googles pushservere, og gøre brug af de push-tokens som er registreret i klient-tabellen (de anvendes ikke af løsningen i dag, men er gemt for at understøtte en fremtidig direkte push-afsendelse udenom AWS).

I forbindelse med deployment kan man vælge at afvikle applikationen som en selvstændig Java applikation (som vist ovenfor), eller man kan pakke den ind i en container (fx Docker), alt efter hvad der giver mest mening i ens driftsmiljø.

For at kunne overtage driften af det kørende driftsmiljø, er det nødvendigt at få en kopi af den aktuelle produktionsdatabase, fx via et SQL backup, som kan etableres i det nye driftsmiljø. Databasen indeholder alle de registrerede MFA klienter, og hvis denne ikke re-etableres, vil alle klienter skulle gen-registreres, hvilket er yderst uhensigtsmæssigt.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden til backenden og kompilere Server API komponenten
* De kan etablere en PUSH gateway i AWS (Via AWS SNS) til at sende push notifikationer, eller tilpasse OS2faktor MFA kildekoden til at sende direkte push beskeder via Apple/Googles push servere.
* De har adgang til en SQL backup af det kørende driftsmiljø, som de kan etablere i en MySQL eller MariaDB database

## Selvbetjening / Administrationsmodul

Brugergrænsefladen ligger i underfolderen ”frontend/ui”.

Bemærk at der også ligger en ”ooapi” folder i ”frontend” folderen. Dette er en kopi af NemID tjensteudbyderpakken, som anvendes til at bruge NemID til registrering af klienter.

Der skal til dette formål være registreret en tjensteudbyderaftale med Nets om brug af NemID. Enten skal den nye driftsleverandør sikre at den ny sådan fremskaffes, eller også skal den nuværende driftsleverandørs aftale overdrages. Til aftalen er knyttet et aftale ID samt et keystore/certifikat.

Konfigurationen af NemID ligger i nemid.properties og pid.properties konfigurationsfilerne i kildekoden, og skal tilpasses med oplysninger om den aktuelle NemID tjensteudbyderaftale.

Når alle modulerne er kompileret, er brugergrænsefladen det også, og man kan starte det op fra kommandolinjen ved at skrive

$ mvn spring-boot:run

Dette afvikler kodebasen “in-place”, hvilket er fint til afprøvning af kildekoden.

Når man kompilerer projektet, dannes en selvindeholdt JAR fil under ”target” folderen, som indeholder den fulde applikation, eksklusiv konfiguration. Man kan kopiere denne applikation over på en server, og så placere konfigurationsfilerne og NemID certifikatet i en konfigurationsfolder ved siden af, fx på denne måde

.

├── config

│   └── application.properties

│   └── pid.properties

│   └── nemid.properties

│   └── nemid-certifikat.pfx

└── os2faktor-frontend.jar

Herefter kan man starte applikationen op ved at skrive

$ java -Dloader.path=”config/” -jar os2faktor-frontend.jar

Man kan tage udgangspunkt i de konfigurationsfiler der ligger i kildekodens ”config” folder, og tilpasse til ens konkrete deployment behov.

Da brugergrænsefladen både udstiller en selvbetjeningsportal og en adminsitratorportal, er der logik i kodebasen som kigger på det DNS/HOST navn som anvendes når man tilgår den kørende applikation. Hvis applikationen tilgås via frontend.os2faktor.dk, så rammer man selvbetjeningsfunktionaliteten, mens man rammer administratorportalen i alle andre tilfælde.

Under udvikling kan man derfor med fordel tilrette sin lokale hosts-fil, så frontend.os2faktor.dk peger på localhost.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden til backenden og kompilere brugergrænsefladen komponenten
* De kan tilgå både selvbetjening og adminsitratorfunktionaliteten i deres lokale udviklingsmiljø
* De har etableret en NemID tjenesteudbyderaftale til produktionsmiljøet

## Websockets

Websocket projektet ligger under “websocket” folderen. Dette projekt er den backend der er knyttet til OS2faktor MFA Windows klienterne. Disse kommunikerer via websockets (HTTPS/443) for at undgå problemer med firewalls, og websocket komponenten er den backend der modsvarer Server API komponenten, men blot til Windows MFA klienterne.

Når alle modulerne er kompileret, er Websocket projektet det også, og man kan starte det op fra kommandolinjen ved at skrive

$ mvn spring-boot:run

Dette afvikler kodebasen “in-place”, hvilket er fint til afprøvning af kildekoden.

Når man kompilerer projektet, dannes en selvindeholdt JAR fil under ”target” folderen, som indeholder den fulde applikation, eksklusiv konfiguration. Man kan kopiere denne applikation over på en server, og så placere en konfigurationsfil i en konfigurationsfolder ved siden af, fx på denne måde

.

├── config

│   └── application.properties

└── os2faktor-websockets.jar

Herefter kan man starte applikationen op ved at skrive

$ java -Dloader.path=”config/” -jar os2faktor-websockets.jar

Man kan tage udgangspunkt i den konfigurationsfil der ligger i kildekodens ”config” folder, og tilpasse til ens konkrete deployment behov.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden til backenden og kompilere websocket komponenten

# Overdragelse af OS2faktor MFA Connectorene

Der er udviklet 2 Connectors til OS2faktor MFA, der gør det muligt at anvende OS2faktor MFA funktionaliteten fra disse produkter på en nem måde.

## OS2faktor MFA via AD FS

Kildekoden til denne Connector findes her

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/connectors/ADFS-Plugin>

Ovenstående Connector er lavet i C#, og er et plugin til AD FS servere, som gør det muligt at anvende OS2faktor MFA som en MFA komponent i Microsoft AD FS.

Da plugins til AD FS altid er versions-specifikke, skal man bygge en version til hhv Windows Server 2012 R2, Windows Server 2016 og Windows Server 2019 når man bygger et nyt release. Dette gøres ved at skifte den .NET reference der ligger i ”adfs-libs” folderen ud mellem hvert byg.

Der er lavet et InnoSetup baseret installer-script, der kan danne en EXE installer. Man kan vælge at gøre brug af dette til at lave en nem installationsmetode til kommunerne, eller man kan bruge et andet værktøj som AdvancedInstaller eller lignende værktøjer.

Hvis man ændrer versionsnummeret i selve DLL filen, så skal man også tilrette de to powershell scripts der ligger i ”Resources” folderen, da de refererer til DLL filerne med versionsnumre. Det anbefales at man undgår at skifte versionsnummeret i DLL filerne, da det gør opdateringen mere besværlig for kommunerne.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden til AD FS connectoren, og kompilere denne
* De kan bygge en EXE installer til at lette installationen af connectoren for kommunerne

## OS2faktora MFA via RADIUS

Kildekoden til denne Connector findes her

<https://github.com/OS2faktor/OS2faktor-MFA/tree/master/connectors/RadiusServer>

Ovenstående Connector er lavet i Java, og baserer sig på TinyRadius frameworket til at lave en simpel pass-through RADIUS server, som kommunerne kan have kørende on-premise.

Connectoren kræver Java 8 (update 151) for at kompilere og afvikle, og kan kompileres vha Maven på følgende måde

$ mvn clean install

Ovenstående bygger en selv-indeholdt JAR fil, som kan deployes sammen med en konfigurationsfil som backenden gør, dvs ved at placere JAR filen sammen med en konfigurationsfolder med konfigurationsfilen

.

├── config

│   └── application.properties

└── os2faktor-radius-server.jar

Herefter kan man starte applikationen op ved at skrive

$ java -Dloader.path=”config/” -jar os2faktor-radius-server.jar

Man kan gøre brug af den eksempel konfigurationsfil der ligger i ”config” folderen i kodebasen. Løsningen skal afvikles on-premise hos den enkelte kommune, og installationen her afhænger af den enkelte kommunes lokale driftssetup.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilgå kildekoden til RADIUS connectoren, og kompilere denne

# Overdragelse af os2faktor.dk domænet

OS2faktor MFA er en multi-tenant løsning – den primære årsag til dette er MFA klienterne, som deployes i diverse app stores. For at undgå at der skal deployes en ”app” per kommune, er disse apps konfigureret til at kommunikere med en fælles backend, som er driftet i OS2 regi for alle tilsluttede kommuner.

Den fælles backend er driftet under os2faktor.dk domænet, og klienterne kommunikerer med

* admin.os2faktor.dk
* frontend.os2faktor.dk
* backend.os2faktor.dk
* websockets.os2faktor.dk (nogle få gamle Windows klienter kommunikerer mod denne)
* websockets2.os2faktor.dk (dette er det nye endpoint som nye Windows klienter kommunikerer mod)

Snitfladen til websockets blev ændret efter den første version af windows klienten blev releaset, og da der ikke er nogen måde at tvinge en opdatering af denne klient, blev løsningen at styre kommunikationen via 2 forskellige DNS endpoints. Slutmålet er at kunne lukke for websockets.os2faktor.dk, men så længe der er PC’ere der stadig kører version 1.0.0 af windows klienten, er dette ikke muligt.

**OBS!** Den nuværende driftsleverandør står som ejer af os2faktor.dk domænet og dette bør overdrages til OS2.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De kan tilrette DNS records for domænet, så de kan pege trafikken mod det nye driftsmiljø

# Opsigelse af aftale og sikre alle data slettes

Den eksisterende driftsaftale på OS2faktor MFA opsiges hos nuværende leverandør, og det sikres at alle data, nøgler m.m., der befinder sig under den nuværende leverandørs kontrol, slettes jf den indgående databehandleraftale.

# Sikre revision hos den nye driftsleverandør

Driften af OS2faktor produkterne (MFA og Login) er under revision for at kunne anvende denne til NSIS anmeldelse. Det bør sikres at den nye driftsleverandør etablerer et driftsmiljø og en tilhørende driftsorganisation som kan overholde kravene til en NSIS revision, og at en sådan revision bliver udført inden overdragelsen af driften, da dette er en forudsætning for den enkelte kommunes egen NSIS revision og anmeldelse af deres løsning til Digitaliseringsstyrelsen.

I forbindelse med en overdragelse af OS2faktor MFA til en anden driftsleverandør, bør den nye driftsleverandør sikre at

* De er bekendt med alle kravene til NSIS revision
* De har etableret alle de fornødne organisatoriske, tekniske og driftsmæssige foranstaltninger, der gør det muligt at gennemføre en NSIS revision
* Der udføres en NSIS revision, og at revisionsrapporten udleveres til kommunerne, så de kan indeholde denne i deres anmeldelse til Digitaliseringsstyrelsen

Ovenstående bør ske før de nye driftsmiljø tages i brug.