Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

Rune Sterten Marhaug
Henrik Hove Eide

Mulighetene med Azure Sentinel

Et proof of concept

Bacheloroppgave i Informatikk med spesialisering i drift av datasystemer
Veileder: Jostein Lund

Mai 2021

NTNU
Kunnskap for en bedre verden

Rune Sterten Marhaug
Henrik Hove Eide

Mulighetene med Azure Sentinel

Et proof of concept

Bacheloroppgave i Informatikk med spesialisering i drift av datasystemer
Veileder: Jostein Lund
Mai 2021

Norges teknisk-naturvitenskapelige universitet
Fakultet for informasjonsteknologi og elektroteknikk
Institutt for datateknologi og informatikk

NTNU

Kunnskap for en bedre verden

Innholdsfortegnelse

- Sluttrapport
- Forstudierapport
- Designrapport
- Driftsrapport

Bacheloroppgave 10

Sluttrapport

Muligheter med Azure Sentinel

Et proof of concept

Av Henrik Hove Eide og Rune Sterten Marhaug

Kontaktinformasjon til Studentene

Henrik Hove Eide – Bachelorstudent, *Informatikk, drift av datasystemer*

Henrieid@stud.ntnu.no - Telefon +47 95252931

Rune Sterten Marhaug – Bachelorstudent, *Informatikk drift av datasystemer*

Runesmar@stud.ntnu.no +47 47446549

Innholdsfortegnelse

Kontaktinformasjon til Studentene 2

1. Forord 4

2. Oppgavebeskrivelse 5

3. Hvordan ble oppgaven løst? 6

3.1 Benyttede metoder og standarder 6

3.2 Bruk av litteratur og Internett 6

3.3 Benyttet programvare 6

3.4 Arbeidsfordeling 8

3.5 Utarbeidet dokumentasjon 9

4. Gjennomføring av prosjektet 10

4.1 Prosess 10

5. Videre arbeid 16

6. Endringer underveis i prosjektet 17

1. Forord

Denne rapporten presenterer resultatene av en bacheloroppgave skrevet av Henrik Hove Eide og Rune Sterten Marhaug i samarbeid med Atea. Oppgaven er avsluttende for studiet Informatikk med spesialisering i drift av datasystemer ved NTNU.

Studentene møtte Atea i oktober 2020 med et klart ønske om å arbeide med IT-sikkerhet. Dette er et tema som er svært aktuelt, spesielt i sammenheng med den økende trusselen for dataangrep som norske bedrifter og organisasjoner står ovenfor. Og med den stadig økende mengden informasjon som er lagret i skyen, er det viktigere enn noen gang at bedrifter bruker investerer de nødvendige ressursene for å kunne sikre miljøet sitt i skyen. Målet med denne oppgaven er derfor å undersøke hvordan skytjenesten Azure Sentinel kan hjelpe bedrifters IT-sikkerhet, og at studentene skal ha et grunnlag for å kunne arbeide videre med sikkerhet.

Vi ønsker å takke Atea som har latt oss bruke de nydelige kontorlokalene deres, og tatt oss godt imot. Spesielt vil vi takke Morten Schjetne som har vært veileder, og Stian André Høydal for å ha bidratt med mye nyttig informasjon som ga oss et solid utgangspunkt for oppgaven. Til slutt vil vi takke Jostein Lund, vår veileder fra NTNU som har bidratt med uvurderlig tilbakemelding og støtte til prosjektet.

Oppgaven har resultert i en driftsrapport som fungerer som en brukerguide. Hensikten vår er at andre skal kunne ta det vi har lært og den løsningen vi har kommet frem til, og bruke det som et utgangspunkt for videre arbeid med Sentinel. Oppgaven dekker ikke alle mulighetene som finnes i Sentinel, men gir et svært solid utgangspunkt, og vi er trygge på at systemet slik som det er presentert vil kunne ha en positiv effekt på enhver bedrifts IT-sikkerhet.

Underveis i prosjektet har vi gjort oss kjent med de ulike komponentene som Sentinel består av, og opparbeidet oss kompetanse innenfor Azure generelt. Vi har fått mye erfaring med relaterte verktøy i Azure slik som Azure AD, Azure Firewall og Azure Logic Apps. Vi lærte tidlig i prosjektet at å sette opp Sentinel og knytte opp datakilder er svært enkelt, derfor har vi fokusert i stor grad på automasjon av systemet, og knyttet det opp mot prinsippet «infrastructure as code». Dette har resultert i et system som kan raskt etableres i nye miljø og kan distribueres til et uendelig antall miljøer ved bruk av Azure DevOps.

2. Oppgavebeskrivelse

Oppgaven går ut på å bygge et «proof-of-concept» basert på Microsoft 365, hvor Microsoft Sentinel benyttes som SIEM-løsning. Å lage et «proof-of-concept» betyr at oppgaven i stor del handler om å utforske hvilke muligheter som finnes i Sentinel, og å lære og formidle «best practices», og til slutt vurdere, basert på det vi har lært, hvordan Sentinel kan påvirke en organisasjons IT-sikkerhet.

Sentinel er et produkt i Microsofts Azure-skyløsning. Et SIEM-system er et system som samler loggdata fra flere ulike kilder, deriblant Microsoft og Windows produkter, på samme plass. Dette gjør at bedriften og dens IT organisasjon kan ha bedre oversikt over aktiviteten i egen infrastruktur og kan jobbe mer proaktivt med kontroll på egne data. Viktige momenter for oppdragsgiver var hvordan Sentinel kan påvirke IT-sikkerheten i en organisasjon, hvordan Sentinel best kan implementeres, og hvordan SOAR-funksjonaliteten i Sentinel muliggjør det å automatisk håndtere sikkerhetshendelser for å spare tid i sikkerhetsarbeidet.

Et hovedfokus for prosjektgruppen ble å forenkle implementeringen av Sentinel i størst mulig grad, med baktanken om at Atea har ønske å kunne tilby en komplett Sentinel-løsning til sine kunder. I denne sammenheng har vi valgt å løse denne delen av oppgaven ved bruk av Azure DevOps pipelines for å kunne drifte Sentinel parallelt i flere miljø. Bruken av dette verktøyet krever at prosjektgruppen retter seg etter arbeidsmetoden DevOps, og prinsippene om kondinuerlig leveranse og kontinuerlig integrasjon (continous delivery og continous integration (CI/CD)).

Slik oppgaven ble tolket konstruerte vi et testmiljø i Azure der vi satte opp Azure Sentinel, med tilhørende tjenester som var nødvendige for testing, utprøving og praktiske demonstrasjoner.

Sentinel har også veldig mange muligheter som ligger utenfor Microsoft 365, men oppgaven ble begrenset til å omfatte Microsoft sine produkter og tjenester og utførelse av oppgaven har derfor konsentrert seg om disse.

3. Hvordan ble oppgaven løst?

3.1 Benyttede metoder og standarder

Prosjektet baserer seg hovedsakelig på teknologier og programvare utgitt av Microsoft. Derfor har vi prøvd i størst mulig grad å rette oss etter «best practices», som er prinsipper og arbeidsmetoder basert på erfaringer gjort av Microsofts egne eksperter. En viktig del av Sentinel «best practices» er å drifte Sentinel som kode. For å oppnå dette har vi tatt i bruk DevOps, og følger prinsippene for CI/CD. Rapportene er skrevet etter dokumentmaler utgitt av NTNU, og bruker APA som referansestil.

3.2 Bruk av litteratur og Internett

Informasjonen og kunnskapen nødvendig for å gjennomføre prosjektet er i størst grad hentet fra Microsoft sin egen dokumentasjon. Under dette har vi Microsoft Docs, Microsoft Developer Forum, Microsoft Tech Community, Azure Sentinel Level 400 Ninja Training og Microsofts egne webinar. Det at nesten all informasjon er hentet fra utvikleren av programvaren vi har brukt, gjør prosjektgruppen trygge på at all informasjon presentert i dokumentasjonen er pålitelig.

3.3 Benyttet programvare

I prosjektet har vi eksklusivt benyttet oss av tjenester og programvare utgitt av Microsoft. Tabellen under viser en oversikt over hvilke tjenester som er tatt i bruk. Det er ingen krav til maskinvare for prosjektet, fordi prosjektet er skybasert.

Programvare og verktøy i Azure	Brukt til
Azure Sentinel	Utgangspunktet og hovedfokuset i prosjektet.
Azure AD P2	Styre brukerrettigheter i Azure og Sentinel, og for å generere loggdata til Sentinel.
Azure Logic Apps	Bygge playbooks i Sentinel.
Azure Virtual Machines	Opprette log collector for å håndtere CEF-logger.

Samhandlingsverktøy	Brukt til
Azure DevOps Repos	Lagring av filer, versjon- og kildekontroll.
Microsoft Teams	Planlegging og gjennomføring av møter.
SharePoint	Lagring og deling av dokumenter.

Verktøy for programmering og skript	Brukt til
Azure DevOps Pipelines	Kontinuerlig levering og integrasjon. Validering av script og konfigurasjonsfiler.
Azure DevOps Resource Manager	Utrulling av ressurser til Sentinel.
PowerShell	Hente data fra Sentinel, sende data fra DevOps. Både ved bruk av ARM og API.
Azure CLI	Powershell klient.

3.4 Arbeidsfordeling

Tabellen under viser hvordan vi har fordelt arbeidet i prosjektet. Både arbeidet og læringen i prosjektet er gjort i samarbeid, men vi har hver for oss spesialisert oss. Rune har hatt fokus på KQL-språket og CEF-logger, mens Henrik har satt opp automasjonsdelen med DevOps og ARM.

Arbeidsfordeling		
Henrik	**Begge**	**Rune**
Azure DevOps	Forstudierapport	Log collector
Pipelines og deployment	Designrapport	Connectors
ARM- og API automasjon	Driftsrapport	Alert rules
	Sluttrapport	Møteinnkallinger
	Møtevirksomhet	Møtereferater

3.5 Utarbeidet dokumentasjon

Dette er en oversikt over all dokumentasjonen som har blitt produsert i løpet av prosjektet.

Rapporter:

Forstudierapport

Designrapport

Driftsrapport

Sluttrapport

Vedlegg i rapporten:

Prosjekthåndbok

Møtereferater

Prosjektpresentasjon

Vedlagt kildekode:

Konfigurasjonsfiler, JSON

Pipeline definisjoner, YAML

Script for deployment, PowerShell

4. Gjennomføring av prosjektet

4.1 Prosess

4.1.1 Tidslinje

Det første møtet i prosjektgruppen ble holdt den 21 oktober 2020 ble det holdt oppstartsmøte for bachelorprosjektet på Ateas kontor i Trondheim. Dette intromøtet ble brukt til å diskutere mulige problemstillinger for bachelorprosjektet og hvordan vi så for oss at prosjektet kunne gjennomføres med tanke på Coronasituasjonen. 23 oktober 2020 ble det oversendt forslag til oppgave fra oppdragsgiver Atea til prosjektgruppen, og det ble raskt enighet om problemstillingen som var utarbeidet og det ble satt en dato for offisiell prosjektstart til 11 januar 2021.

I januar 2021 ble prosjektet offisielt påbegynt. På det første møte kom veilederne med innspill til hvordan prosjektet skulle gjennomføres, og hvilke krav og forventninger som ligger i et slikt prosjektarbeid. Resten av januar bruktes tiden til å få oversikt over temaet, og å lage utkast til forstudierapporten. Det ble også satt opp et testmiljø hos oppdragsgiver som skulle benyttes til praktisk gjennomføring av selve prosjektet og dette fikk vi tilgang til i starten av februar.

Når vi fikk tilgang til Azure hadde vi også ferdigstilt førsteutkast til forstudierapporten samt begynt å gjøre forberedende arbeid til designrapporten. Prosjektgruppen gjennomførte også Sentinel «ninja-training», som er et kurs utgitt av Microsoft.

Mot slutten av februar åpnet Atea kontoret opp og prosjektgruppens første arbeidsdag med fysisk oppmøte var 19. februar. Da ble det holdt et statusmøte med oppdatering om designrapport. Samtidig som vi arbeidet med designrapporten og planleggingen av hvilket sluttresultat vi ønsket, ble det også satt opp ressurser som var nødvendige for å løse prosjektet som DevOps og log forwarder. Vi forsøkte også å lage honeypots for å kunne generere data via Azure Firewall, men dette utgår fra det endelige produktet.

Starten og midten av mars ble brukt på å sette seg inn i hvordan man arbeider med DevOps og hvordan man får koblet alt dette sammen med Azure og Sentinel. Vi undersøkte hvordan man bruker Sentinel sin API for automasjon, og lærte oss å skrive og kjøre spørringer mot data i Sentinel.

Slutten av mars ble brukt til ferdigstillelse av designrapport etter tilbakemeldinger fra veiledere. Siden prosjektet ikke hadde et klart definert mål, ble det mye prøving og feiling underveis, men dokumentasjonen og gjennomføringen av prosjektet gikk relativt greit.

April gikk i hovedsak til løsning av to store problemer vi støtte på, da resten av det praktiske arbeidet i stor grad var ferdigstilt, og vi var avhengige av å løse de to problemene vi hadde støtt på for å kunne fortsette arbeidet. Da disse viste seg og ikke være konfigurasjonsfeil, men et lisensproblem i Azure AD og et rettighetsproblem i Azure som skapte problemene.

Slutten av april gikk med til finpussing og ferdigstillelse av alle rapportene og dokumentasjonskriteriene i prosjektet, samt å lage en praktisk demonstrasjon av hvordan deler av prosjektet fungerer.

Starten og midten av mai er da prosjektet skal ferdigstilles og bacheloren leveres, så her er siste finpuss på alt av dokumentasjon, samt å lage presentasjon og presentere sluttresultatet av prosjektet for oppdragsgiver og andre interesserte.

4.1.2 Tanker rundt prosessen

Atea har vært flinke til å støtte opp om prosjektet vårt, vi har fått hjelp, tilganger og lisenser etter hvert som vi har fått behov for det. Vi har også fått muligheten til å kontakte andre i bedriften for samtaler, hjelp og støtte. Alt i alt har dette vært veldig positivt.

Atea har også gitt oss tilgang til et bra arbeidsmiljø både fysisk på kontoret og digitalt. Vi har vært heldige å få unngå begrensninger på tilgang eller programvare. Dette har gjort at vi har kunnet utforske muligheter innenfor oppgaven som ble gitt oss uten begrensninger.

Tilbakemeldingene og responsen fra veilederne i møtene vi har hatt om prosjektet og rapportskrivingen underveis har vært god. Dette har i stor grad hjulpet oss med å forbedre resultatet av prosjektet. Det har også vært gode svar på spørsmål knyttet til den formelle gjennomføringen.

Det å jobbe rundt en problemstilling uten et klart definert sluttresultat, har gitt oss en unik mulighet til å lære noe både om samarbeid, og hvordan man kan bruke hverandre til å komme frem til den beste løsningen sammen. Det har vært fint å stå fritt til å tolke oppgaven sånn vi vil. Samtidig kommer dette delvis som et resultat av at oppdragsgiver har vært utydelig på hva ønsket med oppgaven var. Vi har derfor måttet komme med egne forslag, tanker og ideer. Vi har fått støtte for de forslagene vi kom med, og det vi har kommet frem til. I tillegg var veileder fraværende grunnet sykdom, til den grad at han var utilgjengelig i den perioden der vi måtte låse fast designet av sluttproduktet.

Kommunikasjonen fra NTNU har også vært varierende i denne prosessen. Dokumentmaler og fremgangsmåter for formelle prosesser har ikke vært tilgjengelig fra en sentral ressurs. For hver rapport i prosjektet har vi hovedsakelig brukt tidligere rapporter som utgangspunkt for våre egne, i tillegg til å mase på veileder.

I starten av prosjektet brukte vi god tid på å legge en plan for prosjektet og fulgte denne på en god måte. Underveis i det praktiske arbeidet ble vi litt for opphengt i det vi selv drev med og mistet den overordnede planen av syne. Dette førte til at vi mistet litt struktur og rytmen i arbeidet ble noe oppstykket. Hadde vi kunnet gått tilbake og endret på noe i ettertid ville vi nok sørget for å holde bedre på strukturen og holdt planleggingen ved like gjennom hele prosjektet. Det er også vanskelig å planlegge hele veien når det er et tema og en læringsprosess vi som studenter også skal gjennom. Vi hadde ikke full oversikt over temaet eller prosessen vi skulle gjennom, og dette bidro også til at det var vanskelig å planlegge det praktiske i prosessen.

4.1.3 Resultat og måloppfyllelse

Det er kommet ønske om å inkludere en evaluering av Sentinel som sikkerhetsverktøy i sluttrapporten. Derfor presenterer vi det vi har lært i dette kapittelet, og håper dette vil være nyttig for de leserne som ønsker å forstå «hva og hvorfor Sentinel» uten å sette seg inn i de tekniske og praktiske delene av å arbeide med Sentinel.

Trusselbilde

Det å formulere et trusselbilde som dekker de fleste norsk bedrifter eller organisasjoner er en utfordring. Men basert på erfaringer hos ansatte i Atea som arbeider med sikkerhet daglig er en fellesfaktor for bedrifter som blir utsatt for cyberangrep manglende to-faktor autentisering, oversikt, innsikt og innsyn i sin infrastruktur. Dersom vi vil bruke et nylig eksempel der Østre Toten kommune ble utsatt ransomware i januar 2021, viste det seg at hackerne hadde hatt tilgang på systemet i en lengre periode, og fått tilgang til både sensitive personopplysninger og backupløsninger.

Tradisjonelle sikkerhetsløsninger, er ofte ikke alltid gode nok på å oppdage trusler der motiverte inntrengere har fått legitim tilgang til et system gjennom phishing. I en slik type angrep er det relativt enkelt for en kyndig angriper å bevege seg lateralt, og forbli uoppdaget fordi bevegelsen og aktiviteten foregår i så små skritt at det ikke setter av noen alarmer.

Gode sikkerhetssystemer er ofte også komplekse, og består av flere separate systemer. Vi snakker her om antivirus og anti-malware, endpoint protection, VPN, intrusion detection/prevention, file integrity monitoring, brannmurer og så videre. I tillegg blir det generert loggfiler fra servere og nettverksutstyr. Alle disse systemene produserer store mengder data, som er vanskelig å alltid ha oversikt over dersom en ikke har gode rutiner og kvalifisert personale som kan tolke informasjonen.

Hva er sentinel?

Sentinel er en skybasert SIEM-løsning utgitt av Microsoft. SIEM er en forkortelse for security information and event management. SIEM er en kombinasjon av security information management (SIM), et system som samler inn loggfiler fra ulike kilder, og security event

manager, et system som bruker ulike analyseverktøy for å forstå loggdata. SIEM-løsninger kombinerer disse to produktene og har muligheten til å både samle inn og analysere loggdata I sanntid.

Hvorfor Sentinel?

Den største fordelen med en skybasert SIEM er at en bedrift ikke trenger investere i maskinvare. Dette har to store fordeler, den første er at det ikke trengs personale for å administrere og drive vedlikehold på fysiske maskiner, den andre fordelen er at en kun betaler for det en bruker, og kan enkelt skalere opp eller ned basert på hvor mye data som blir generert til et hvert tidspunkt. I tillegg garanterer Microsoft Azure 99,9% tilgjengelighet.

Microsoft har gjort det svært enkelt å ta i bruk Sentinel. For over hundre tjenester og produkter er det så enkelt som å trykke på en knapp. Dette inkluderer Microsoft 365 tjenester som Office, Teams, SharePoint, Defender og Intune. I tillegg har vi produkter fra blant annet Citrix, Cisco, G-suite og AWS med flere. Denne listen vokser stadig, og Sentinel er fremdeles under kontinuerlig utvikling.

Alle datakilder kommer med forhåndsdefinerte regler som er utviklet av Microsofts sikkerhetseksperter. Disse reglene kan aktiveres med et enkelt klikk, og vil umiddelbart sette i gang med å analysere informasjonen som kommer inn til Sentinel. Basert på reglene vil Sentinel automatisk varsle dersom det er hendelser av interesse i dataen.

Vi er trygge på at dersom vi fikk i oppgave å installere Sentinel i en bedrift, ville vi hatt det oppe innen maksimalt to dager, uavhengig av hvor mange systemer som skal kobles opp. Gjennom vårt arbeid med Azure DevOps har vi også demonstrert at det er mulig å rulle ut Sentinel til flere miljø samtidig, og vi kan drifte samtlige systemer fra en sentral kodedatabase.

Hva er SOAR?

Tradisjonelt vil varslinger generert av sikkerhetssystemer kreve at en sikkerhetsanalytiker er tilgjengelig for å tolke og eventuelt reagere på hendelsen. Med Sentinel sin SOAR kapabilitet, som står for «Security Orchestration, Automation and Response», kan dette arbeidet i stor grad

automatiseres.

Grunnlaget for SOAR er såkalte playbooks, som er et regelsett for hvilke handlinger som skal gjøres når en alarm går av. Playbooks er basert på Azure Logic Apps, og tilbyr hundrevis av muligheter for automasjon. Som eksempel har vi skrevet en playbook som oppdager mistenkelige påloggingsforsøk, deretter vil brukerkontoen automatisk bli sperret ute, og en e-post blir sendt til en systemadministrator. Det er ofte snakk om «alert fatigue» i sikkerhetskretser, fordi det stadig er noe å ta tak i vil hvert enkelt varsel bli oppfattet som mindre og mindre viktig. Derfor hjelper Sentinel de sikkerhetsansatte med å redusere tiden de bruker på oppgaver som er preget av rutine.

Avanserte verktøy i Sentinel

Sentinel lener seg også i stor grad på maskinlæring for å oppdage uregelmessigheter og trusler som fort kan fly under radaren til andre sikkerhetssystemer. Dette gjøres både i form av UEBA (user and entity behavior) og spesielle analytic rules. Sentinel er altså selvlærende, og vil etter kort tid kunne forstå en normalsituasjon i miljøet det er installert i, og vil automatisk varsle dersom det oppdager trusler. Dette er spesielt relevant for å beskytte seg mot såkalte multistage angrep.

Mange av varslene som Sentinel karakteriserer som «anomalies» er ikke nødvendigvis faktiske trusler. Men de kan være et godt utgangspunkt for å proaktivt jakte etter sikkerhetstrusler. Sentinel gjør det enkelt å grave i de store datamengdene med hunting rules, Hunting rules hjelper deg å spørre de riktige spørsmålene med forhåndsdefinerte regler, og lar deg skrive dine egne for å undersøke om det er hendelser av interesse i systemet. Sammen med hunting rules har vi Notebooks. Dette er dokumenter som kan en kan kjøre kode i, og visualisere resultatene av koden sanntid. Med Notebooks kan du enkelt dokumentere prosessen din for jakting, og dele den med andre slik at de kan kjøre samme kode i sitt eget miljø. Dette faller sammen med at Sentinel Sentinel et stort samfunn av utviklere som aktivt deler kode, spørringer og regler.

Konsekvenser for bedrift

Dersom en bedrift skulle ta i bruk Sentinel vil det bety store endringer for sikkerhetsarbeidet i bedriften. Mye av sikkerhetsarbeidet kan skyves ned til et lavere nivå, fordi du trenger lavere kvalifisert personell til å gjøre jobben enn tidligere, noe som frigjør ekspertene til å gjøre mer avansert sikkerhetsarbeid.

Svakheten til Sentinel er at systemer skalerer med mengden informasjon det har tilgjengelig. Dette gjør også kostnadene, siden du betaler en fast sum per gigabyte. Derfor må en bedrift som velger å ta i bruk Sentinel gå «all-in», og bruke tid på å lære og bygge kompetanse på systemet før det kan brukes mest mulig effektivt.

5. Videre arbeid

Vi repeterer anbefalingen om videre arbeid fra driftsrapporten.

Siden dette arbeidet er et «proof-of-concept» har vi oppdaget at er det mye videre arbeid som er mulig. Dette prosjektet kan karakteriseres som et «minimum viable product» og har rom for videreutvikling. Og med tanke på at oppdragsgiver kanskje er interessert i å selge Sentinel-løsninger som produkt til kundene sine, har vi vi følgende forslag for videre arbeid.

1. Automatisk hente rules fra en dev workspace Å opprette et workspace som blir brukt for å teste nye regler vil la ansatte utforske mulighetene som ligger i Sentinel videre uten at det påvirker et aktivt Sentinel workspace.
2. Workbooks og notebooks Workbooks og notebooks er nyttige verktøy som vi har sett lite på i dette prosjektet.
3. MSSP tenant. MSSP tenant er en spesiell tenant som kan opprettes under en subscription, og gjør at Atea kan aktivere Sentinel for en kunde, uten at de får tilgang til regler og konfigurasjon som Atea kommer til å bruke mye tid på å utvikle. Queries, playbooks og workbooks kan overføres mellom tenants med bruk av Lighthouse. Rules er planlagt å fungere i nær fremtid.

6. Endringer underveis i prosjektet

I utgangspunktet for prosjektet var tanken at vi skulle forsøke å drifte Sentinel i et miljø. Dette ville betydd at vi måtte bruke mye tid på å sette opp andre systemer som ikke er direkte relatert til Sentinel. Spesielt gjelder dette Microsoft 365 produkter som Office og Intune. Det viste seg tidlig at ikke bare var dette enkelt, men vi var på vei inn i en blindgate. Etter å ha satt opp miljøet er det ikke så mye igjen å gjøre. Vi konkluderte, i samtale med veileder, at å bruke mye tid på andre systemer ikke var relevant for oppgaven. Vi ønsket heller ikke å bruke tid på å skrive våre egne regler for systemet, som originalt planlagt, fordi Microsofts eksperter har allerede laget et bredt spektrum av forhåndsdefinerte regler, det var derfor nok for oss å forstå hvordan regler skrives og aktiveres. Derfor endte vi opp med å fokusere mye på automasjonen av Sentinel, og oppdage hvordan det kan opereres på stor skala.

Forstudierapport

Muligheter med Azure Sentinel

Et proof of concept

Av Henrik Hove Eide og Rune Sterten Marhaug

Endringslogg

Trondheim, 06/05/2021

Å komme tilbake til denne rapporten, som originalt ble skrevet i jan/feb 2021 for å gjøre klar til endelig innlevering, kommer det frem at dette prosjektet har gjennomgått betydelige endringer underveis. Spesielt i ettertid virker prosjektmålene vage og bakgrunnen for prosjektet er tynn. Dette er en konsekvens av at prosjektgruppen ikke hadde den nødvendige forståelsen for teknologien, verktøyene og dagligsituasjonen som var nødvendige å opparbeide seg for å løse oppgaven, som i utgangspunktet allerede var åpen for tolkning.

Vi har valgt å la forstudierapporten stå som den er, og refererer leseren videre til diskusjonen om prosjektets gang i sluttrapporten for innsyn i hvorfor det er gjort endringer i prosjektet.

Dato	Beskrivelse	Forfatter
12.02.21	Presisert mål og bakgrunn for prosjektet	Rune, Henrik
06.05.21	Stilsetting, referanser, rettskriving	Henrik
16.05.21	Ferdigstilling av dokument	Rune, Henrik

Terminologi

Proof of concept	En demonstrasjon, i form av et pilotprosjekt som kan verifisere det praktiske potensialet til en ide.
Azure Sentinel	Microsoft sin skybaserte SIEM-løsning.
SIEM	Security Information and Event Management, system som konsoliderer og behandler loggdata fra flere kilder.
SOAR	Security Orchestrated Automatic Response, et system som kan automatisk behandle sikkerhetshendelser i et system.
Microsoft Azure	Microsoft sin skytjeneste.

1. Introduksjon – hensikten med dokumentet

Denne forstudierapporten er skrevet i som del av en bacheloroppgave i Informatikk med spesialisering i drift av datasystemer. Oppgaven blir gjennomført i samarbeid med Atea.

Med utgangspunkt i forstudierapporten skal både prosjektgruppen, som består av studentene Henrik Hove Eide og Rune Sterten Marhaug, i samarbeid med oppdragsgiver Atea, få en samstemt oppfatning av hva dette prosjektet handler om. Det blir avtalt hvilke resultater og hvilke målsetninger som ønskes oppfylt av prosjektet, hva kostnadene av prosjektet vil være og hvilke risikoer som kan virke negativt på prosjektet. Denne rapporten danner uansett et beslutningsgrunnlag for å avgjøre om prosjektet skal gjennomføres.

Denne forstudierapporten kan anses som en kontrakt eller en avtale mellom oppdragsgiver og de som skal utføre arbeidet, og vi må da sørge for at det inneholder de opplysningene som er nødvendige for at begge parter er trygge på hva de har avtalt.

Hoveddelene av denne rapporten er bakgrunn og mål for prosjektet, presentert i kapittel 2 og 3. I kapittel 2 prøver vi å forklare motivasjonen for prosjektet, og hva prosjektet ønsker å undersøke. På dette grunnlaget utarbeider vi i kapittel 3 konkrete mål som vi ønsker å få oppfylt. Deretter gjør vi en rask risiko og kost-nytte analyse, før vi avslutter med formelle krav og rammebetingelser som ligger til grunn.

Innhold

	Terminologi		20
1.	Introduksjon – hensikten med dokumentet		20
2.	Bakgrunn for prosjektet		22
	2.1	Beskrivelse av problemer og behov	22
	2.2	Kort om dagens systemer og rutiner	23
3.	Prosjektmål		23
	3.1	Effektmål	23
	3.2	Resultatmål	24
	3.3	Prosessmål	24
	3.4	Prosjektets omfang	24
	3.5	Prosjektets milepæler og hovedaktiviteter	24
4.	Interessenter og suksessfaktorer		26
	4.1	Interessentanalyse med suksessfaktorer	26
5.	Rammebetingelser og informasjonsbehov		27
	5.1	Økonomiske rammer	27
	5.2	Tidsmessige rammer	27
	5.3	Informasjonsbehov	27

6. Risikoanalyse 28
6.1 Risikotabell 28
7. Kost/nytte-analyse 32
7.1 Kvantifiserbar og ikke-kvantifiserbar nytte. 32
7.2 Bortfall av direkte kostnader 33
8. Retningslinjer og standarder 33
8.1 Krav til dokumentasjon 34
8.2 Krav til standarder og metoder 35
8.3 Endringshåndtering 35
9. Prosjektorganisering 36
10. Anbefaling om videre arbeid 36
11. Referanser 36

2. Bakgrunn for prosjektet

Oppdragsgiver Atea ønsker at prosjektgruppen skal bygge et Proof of Concept basert på Microsoft 365 med Microsoft Sentinel som SIEM-løsning. Microsoft Sentinel er et verktøy i skytjenesten Microsoft Azure som samler inn logger og hendelser fra programvare og enheter i et nettverk, og analyserer aktiviteten i nettverket i sanntid, på jakt etter sikkerhetstrusler. Videre tillater Sentinel en administrator å sette opp automatisk hendelseshåndtering for å effektivisere sikkerhetsarbeidet. Atea er interessert i å bygge kompetanse på- og se hvilke muligheter som finnes i Microsoft Sentinel.

a. Beskrivelse av problemer og behov

Det har de siste årene vært en økning i profesjonelle aktører som høster høy profitt ved å drive datakriminalitet. Både små og store bedrifter, samt offentlige institusjoner blir rammet. Generelt er det mangel på 2-faktor, logging, oversikt og en helhetlig arkitektur som gjør at bedrifter og kommuner går på en smell. En del av løsningen kan være å bruke en SIEM-

programvare.

b. Kort om dagens systemer og rutiner

Siden dette prosjektet ikke retter seg direkte mot en konkret kunde eller case, har vi valgt å ta utgangspunkt i en generell trusselbeskrivelse. Dette er uproblematisk fordi trusselbildet og utfordringene til de aller fleste norske bedrifter er den samme med tanke på IT-sikkerhet. Norske bedrifter og offentlige foretak opplever i økende grad angrep fra cyberkriminelle som høster store inntjeninger ved å utnytte svak eller manglende IT-sikkerhet eller utnytter samfunnets manglende oppmerksomhet rundt farene for «phishing». Kripos sier også at de opplever en økning i andelen ressurssterke og kompetente angripere[1], noe som gjør denne oppgaven desto mer aktuell.

Vi har per dags dato lite innsyn i hva den gjennomsnittlige bedriften har av sikkerhetssystemer, men vi håper å få en bedre forståelse for dette i løpet av prosjektet.

3. Prosjektmål

Dette kapittelet beskriver mål for prosjektet. Målene vil legge grunnlaget for både gjennomføringen- og evalueringen av prosjektet. Målene er viktige som styringsmidler i prosjektet, og er utarbeidet sammen med veileder. Likevel er mange av målene åpne, fordi dette er et Proof of Concept er vi enda ikke helt sikre på hvordan det ferdige produktet vil være utformet.

a. Effektmål

Dersom det ferdige systemet skulle blitt tatt i bruk av en organisasjon, har vi følgende mål for systemet:

- Øke den generelle IT-sikkerheten i organisasjonen.

[1] (Vollan, 2020)

- Videre styrke IT-sikkerheten ved å kunne aktivt jakte etter trusler.
- Forenkle IT-sikkerhetsarbeidet ved å samle all overvåkning og logginnhenting i ett system.
- Frigjøre arbeidsressurser ved å automatisk identifisere og behandle sikkerhetstrusler.

1.2 Resultatmål

- Systemet skal automatisk oppdage trusler i bla. Azure AD og O365.
- Systemet skal automatisk håndtere trusler basert på en *playbook*.
- Systemet skal være skalerbart, slik at nye datakilder enkelt kan tilføyes.

1.3 Prosessmål

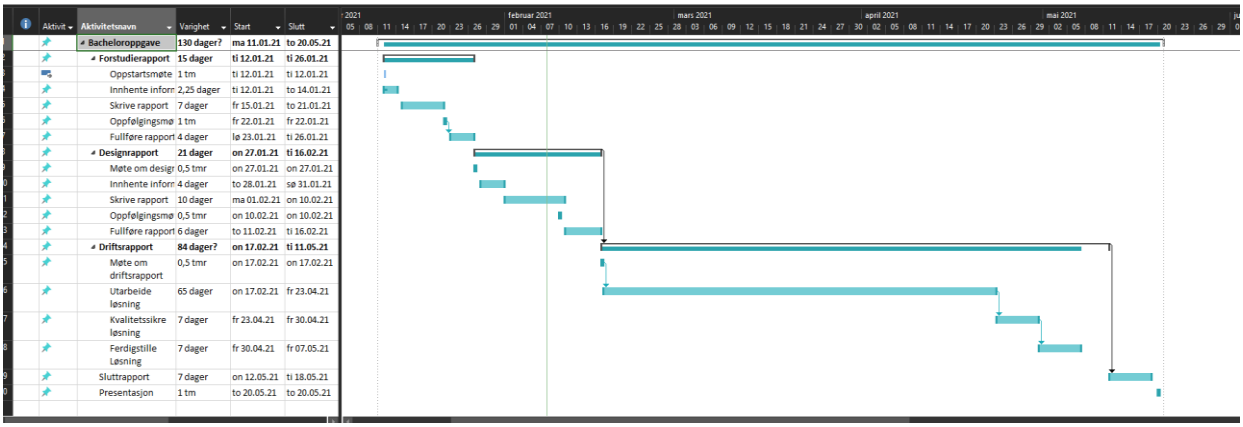
- Ha kompetanse på de ulike sikkerhetsløsningene som tilbys i Microsoft 365.
- Ha forståelse for den daglige situasjonen i en bedrift, og deres krav til sikkerhetsløsninger.
- Mestre de formelle aspektene ved prosjektarbeid, møteinnkalling, rapportskriving og dokumentasjon.
- Oppnå karakteren A på prosjektet.

b. Prosjektets omfang

Vi går inn i prosjektet med lite forhåndskunnskap, prosjektets omfang må derfor være fleksibelt, slik at vi kan få til best mulig resultat. Vi skal derimot forholde oss tett til effektmålene vi har satt. Dette betyr at vi ønsker å unngå å kaste bort tid på verktøy og funksjoner som ikke direkte angår Sentinel, som er prosjektets hovedfokus. Det kan derimot bli nødvendig å ta i bruk andre produkter for å utvide funksjonaliteten til - eller teste Sentinel, dette vil da måtte vurderes i forhold til relevans og tidskostnader i hvert enkelt tilfelle.

c. Prosjektets milepæler og hovedaktiviteter

Et Gantt-diagram er opprettet i Microsoft Project for å illustrere hvordan vi ønsker å fordele arbeidet i prosjektet.



Aktivit	Aktivitetsnavn	Varighet	Start	Slutt
	Bacheloroppgave	130 dager?	ma 11.01.21	to 20.05.21
	Forstudierapport	15 dager	ti 12.01.21	ti 26.01.21
	Oppstartsmøte	1 tm	ti 12.01.21	ti 12.01.21
	Innhente inform	2,25 dager	ti 12.01.21	to 14.01.21
	Skrive rapport	7 dager	fr 15.01.21	to 21.01.21
	Oppfølgingsmø	1 tm	fr 22.01.21	fr 22.01.21
	Fullføre rapport	4 dager	lø 23.01.21	ti 26.01.21
	Designrapport	21 dager	on 27.01.21	ti 16.02.21
	Møte om desigr	0,5 tmr	on 27.01.21	on 27.01.21
	Innhente inform	4 dager	to 28.01.21	sø 31.01.21
	Skrive rapport	10 dager	ma 01.02.21	on 10.02.21
	Oppfølgingsmø	0,5 tmr	on 10.02.21	on 10.02.21
	Fullføre rapport	6 dager	to 11.02.21	ti 16.02.21
	Driftsrapport	84 dager?	on 17.02.21	ti 11.05.21
	Møte om driftsrapport	0,5 tmr	on 17.02.21	on 17.02.21
	Utarbeide løsning	65 dager	on 17.02.21	fr 23.04.21
	Kvalitetssikre løsning	7 dager	fr 23.04.21	fr 30.04.21
	Ferdigstille Løsning	7 dager	fr 30.04.21	fr 07.05.21
	Sluttrapport	7 dager	on 12.05.21	ti 18.05.21
	Presentasjon	1 tm	to 20.05.21	to 20.05.21

Prosjektets milepæler er:

Ferdigstilling av forstudierapport: 26. januar

Ferdigstilling av designrapport: 16. februar

Ferdigstilling av driftsrapport: 11. mai

Levering av prosjektet: 20. mai

Presentasjon av prosjektet: 28. mai

4. Interessenter og suksessfaktorer

a. Interessentanalyse med suksessfaktorer

Interessentanalyse brukes for å identifisere behovene til de ulike interessentene for prosjektet. Her har vi listet opp hva hver interessent ønsker å få ut av prosjektet, og hva de bidrar med.

Interessent	Suksessfaktorer	Bidrag til prosjektet
Eksterne		
- **Atea**	At prosjektgruppen har: God forståelse av sikkerhetsproduktene i Azure. Forståelse av bedrifters behov. Dokumentasjon med høy kvalitet og kunnskapsoverføring.	Ekspertise på fagområdet. Testmiljø/Lab Forståelse av dagligsituasjonen.
Interne		
- **Prosjektgruppen**	Vellykket produkt og prosjektrapport.	Ansvar, utfører prosjektarbeidet og sørger for kommunikasjon mellom interessentene.
- **Veileder**	Et ryddig og godt gjennomført prosjekt. God kommunikasjon.	Kunnskap, veiledning. Godkjenning av prosjektet.

5. Rammebetingelser og informasjonsbehov

a. Økonomiske rammer

Atea dekker kostnadene av lisenser og testmiljø. Derfor er det ikke knyttet noen direkte kostnader til gjennomføringen av prosjektet.

b. Tidsmessige rammer

Prosjektet skal være ferdigstilt innen 20. mai 2021. Dette inkluderer både sluttproduktet og tre rapporter.

Det er antatt at begge medlemmene i prosjektgruppen skal bruke 500 timer hver på arbeidet.

Det skal holdes en presentasjon av prosjektet etter leveringsdato.

c. Informasjonsbehov

Prosjektgruppen ser ikke et tydelig behov for ekstern informasjon utenom veiledning fra veileder og styring fra oppdragsgiver. På den andre siden har både oppdragsgiver og veileder behov for jevnlig informasjon fra studentene som gjennomfører prosjektet. Dette er for å sikre god kommunikasjon. Oppdragsgiver bruker informasjonen for å kunne styre prosjektet mot det sluttproduktet de ønsker. Veileder har behov for informasjon for å kunne overse at frister blir holdt og at nødvendig dokumentasjon produseres.

Av grunnene ovenfor har vi lagt opp til at det skal holdes møter hver andre uke. Det er også åpent for uhøytidelig kommunikasjon på e-post og chat. På denne måten sikrer vi informasjonsbehovet til alle parter i prosjektet.

6. Risikoanalyse

Risikoanalyse er en studie av risiko for å få innsikt i hvilke hendelser som kan skje, hvorfor de skjer og hvilke konsekvenser det vil få dersom de inntreffer. Vi bruker en skala fra 1 til 12 hvor summen har følgende betydning.

- 1-3 tilsier svært sjelden sannsynlighet for at det inntreffer og ubetydelig konsekvens.
- 4-6 tilsier sjelden sannsynlighet og mindre alvorlig konsekvens.
- 7-9 tilsier at det skjer ofte og at konsekvensene vil være alvorlig
- 10-12 tilsier at det skjer svært ofte og at konsekvensene kan være svært alvorlige eller i verste fall fatale.

a. Risikotabell

Svært ofte	5 (3)	10 (5)	15	20	25 (12)
Ofte	4	8 (4,5)	12 (7)	16 (8,9,10)	20
Innimellom	3 (1,2)	6	9(5)	12	15
Sjelden	2	4	6	8	10
Aldri	1	2	3	4	5
	Ikke alvorlig	Mindre alvorlig	Alvorlig	Kritisk	Svært kritisk

	Hva kan gå galt?	S	K	R	Forklaring	Titltak
1	Feilrapportering	3	1	3	Feilrapportering eller false positives er et minkende problem i Sentinel og vil vanligvis ikke ta lang tid å nøste opp slik som systemene i Azure er bygget opp. Konsekvensene blir derfor ikke store, og ikke tidkrevende.	Ta i bruk Sentinels playbooks og ML som lærer kontinuerlig for å minimere antallet false positives i bedriften.
2	Kortvarig sykdom i prosjektgruppen	3	1	3	En eller flere av medlemmene i prosjektgruppen blir syke for en kort periode og dette fører til redusert kapasitet for arbeid i prosjektet i en begrenset periode.	Under de nåværende omstendigheter bør man følge smittevernstiltak, og unngå unødvendig nærkontakt for å holde seg frisk.
3	Nedetid på sentrale tjenester	1	5	5	Med skybaserte tjenester som Azure, vil man kunne oppleve nedetid, og derfor ikke ha tilgang til filer, programmer og tjenester man trenger for å gjøre jobben sin effektivt. Microsoft sine tjenester har oppegaranti på 99,98% og det er derfor svært sjelden og vedlikehold vil være planlagt på forhånd.	Gjør verktøy tilgjengelig offline, slik at en kan jobbe med prosjekter/dokumenter offline. Eventuelt ha en reserveløsning for å ivareta produktiviteten i bedriften.
4	Feil konfigurasjon	2	4	8	Ved feil konfigurasjon kan systemet blokkeres eller rapportere tilgang til systemer som brukes av ansatte for å gjennomføre deres arbeid.	Automasjon av konfigurasjon kan minimere muligheten for menneskelige feil og sørge for at alle får de rettighetene

					ved riktig opplæring skal ikke dette være et problem, og det bør la seg løse dersom man oppdager at dette er tilfellet. ikke veldig alvorlig kan tape litt effektiv arbeidstid avhengig av hvilke tjenester som blir blokkert av systemet, men det bør relativt lett og raskt la seg løse.	de skal ha med en gang, man kan også gå inn manuelt og sjekke at alt fungerer for ulike grupper i organisasjonene etter installasjon da man sjelden eller aldri vil ha behov for individuelle rettigheter, men de vil være gruppebasert.
5	Problemer med utstyr / lab / systemer	2	5	10	Nå som det er Covid-19 utbrudd og alle kontorer er stengt kan det oppstå problemer med testmiljø og tilgang til disse fra oppdragsgivers side. Det skjer mest sannsynlig ikke, men konsekvensene kan påvirke prosjektet dersom noe skulle skje.	Ha en god dialog med oppdragsgiver og sørg for å få rett tilgang til rett tid og at man har en klar kommunikasjon for å unngå misforståelser og for å sørge for at ting blir som begge parter forventer.
6	Innbrudd / uønsket tilgang til sentinel og loggene der?	1	12	12	Dersom noen uønskede skulle få tilgang til Sentinel og informasjonen som rapporteres kan dette være svært ødeleggende. Skal ikke skje og hvert fall ikke regelmessig hvis i det hele tatt, det er likevel en risiko med et slikt system.	Sørg for at bare de som må har tilgang til Sentinel, to faktor autorisering og sikre passord, samt pålogging bare fra bedriftens enheter bør være et krav for å minimere tilgang fra uønskede til såpass kritiske komponenter.
7	Menneskelig svikt?	2	8	16	Hvis man gjør en feil i konfigurasjonen av systemet, kan dette få svært alvorlige konsekvenser.	Sørg for kvalitetssikring av arbeidet som blir gjort og ha flere på samme ansvarsområde som ser over

						endringer og konfigurasjoner som blir gjort slik at man i større grad kan oppdage feil og risikoer.
8	Tap av prosjektfiler	2	8	16	Ved tap av prosjektiler (Forstudierapport, designrapport, driftsrapport, sluttrapport eller script) som utvikles i forbindelse med prosjektet vil det kreve mye merarbeid for å produsere disse på nytt og det vil derfor være kritisk å miste noen av disse.	Ha flere versjoner, bruk skylagring eller ha filene lagret på flere enheter slik at risikoen for å miste filene er minimale og at man i så tilfelle mister bare de nyeste delene av dokumentet.
9	Langvarig sykdom	1	16	16	En eller flere av medlemmene i prosjektgruppen blir syke for en lengre periode og dette fører til redusert kapasitet for arbeid i prosjektet i en lengre periode, og man vil da mest sannsynlig ikke kunne gjennomføre prosjektet.	Under de nåværende omstendigheter bør man følge smittevernstiltak, og unngå unødvendig nærkontakt for å holde seg frisk.
10	Samarbeidsproblemer eller brudd i prosjektgruppen	1	25	25	Dersom det skulle bli problemer i gruppen eller at en av medlemmene ikke jobber med prosjektet vil dette få fatale konsekvenser for prosjektet. Dette er svært lite sannsynlig.	Sørge for god kommunikasjon i gruppen og at man løser eventuelle uenigheter på en god måte. Slik vil de involverte fortsatt være motivert for å jobbe videre med prosjektet.

7. Kost/nytte-analyse

Vanligvis vil det være nødvendig å gjennomføre en kost/nytte-analyse for å ha et tallgrunnlag for hvorvidt prosjektet skal gjennomføres. Her blir effektmålene brukt for å måle den forventede effekten av prosjektet. Med kvantifisering mener vi at vi tallfester nytten, som oftest i kroner. Dette er midlertidig svært vanskelig med tanke på at vi skal bygge et Proof of Concept. Vi skal med andre ord kun se på mulighetene Sentinel kan tilby, og vet ikke hvilke kostnader det ferdige produktet vil ha. Derfor utgår dette punktet fra vår rapport. Vi kan derimot gjøre oss noen tanker rundt nytten til prosjektet.

a. Kvantifiserbar og ikke-kvantifiserbar nytte.

Effektivisering:

Basert på effektmålene vil det ferdige systemet kunne frigjøre tid for de sikkerhetsansatte, dette er fordi Sentinel i stor grad automatiserer mange funksjoner som kan være tidkrevende. Dette vil frigjøre ressurser som kan rettes mot mer proaktivt sikkerhetsarbeid. Ifølge en rapport utarbeidet av Forrester på bestilling fra Microsoft, kan Sentinel redusere andelen falske positiver som må etterforskes med opp til 79%[2].

Skalerbarhet og forutsigbarhet i kostnadene:

Å beregner hvor mye data og lagringsplass man trenger er en stor utfordring. Mengden data som kommer inn per dag i løpet av et år kommer til å variere. Det bedriftene i Forrester rapporten referert over, var at en enten måtte begrense mengden data fra Azure og Sentinel eller å risikere at lagringsplass stod ubrukt og ubenyttet. Med Sentinel i Azure kan man skalere dette etter behov og disse frustrasjonene er ikke lenger en ting. Dette medfører mindre frustrasjon, lavere kostnader og større forutsigbarhet for alle involverte parter.

MTTR ned fra timer / dager til minutter med sentinel, Dette henger sammen med punktet over, fordi man får bedre oversikt med bruk av Sentinel går MTTR (Mean Time to Repair) tiden ned fra timer eller dager til minutter etter at man har fått rapportert inn en hendelse. Dette frigjør personell som ellers ville vært opptatt med vanskelige nøstingsoppgaver og

[2] (Forrester, 2020)

sparer de samme ansatte for en stor mengde frustrasjon.

Man får samlet all dataen man trenger på et sted og man er derfor tryggere på hvilke handlinger som kreves i hvilke situasjoner med sentinel.

Oppdateringer og patcher som før måtte gjøres manuelt gjøres nå automatisk og man får det gratis. Når man skulle oppgradere, oppdatere eller patche en tradisjonell SIEM løsning måtte man koordinere med mange eksperter innenfor de forskjellige feltene med lagrings ekspert på løsningen, sikkerhetseksperten også videre med microsoft får man alle oppdateringer og patcher automatisk og det er kontinuerlig forbedring pluss at man får det gratis!

b. Bortfall av direkte kostnader

Her beregnes for eksempel drifts- og forvaltnings kostnader ved de systemene som vi forventer skal erstattes av det nye systemet. Bortfall av direkte kostnader er en form for nytte og tas med på pluss siden.

Hvis vi ser dette i forhold til kvantifiserbar nytte som vi omtalte i forrige kapittel, så vil vi oppdage at skillet mellom det vi kaller kvantifiserbar nytte og det vi kaller bortfall av direkte kostnader ikke er så absolutt.

Redusert rentetap som vi kalte kvantifiserbar nytte kunne vi for eksempel gjerne se på som bortfall av direkte kostnader. Det viktigste her er imidlertid ikke hvilke overskrifter vi gir de enkelte nyttefaktorene. Det viktigste er at vi får med alle og at vi ikke tar med noen flere ganger. (Altså ikke reduserte rentekostnader både som kvantifiserbar nytte og som bortfall av direkte kostnader).

Innføringen av Sentinel kan gjøre at flere juniorer kan håndtere sikkerhetstrusler noe som frigjør dyr arbeidskraft og samtidig kan redusere lønnskostnaden til et SOC-team.

8. Retningslinjer og standarder

I dette kapitlet skal vi kortfattet ta med de retningslinjene og standardene som prosjektet må forholde seg til. Det gjelder for eksempel:

a. Krav til dokumentasjon

I dette prosjektet skal det utarbeides følgende dokumenter.

Forstudierapport	29. Januar
Designrapport	16. Februar
Driftsrapport	11. Mai
Sluttrapport	18. Mai
Presentasjon	28. Mai

Alle dokumenter gjennomgås med veileder med et utkast, minst en uke før fristen. Dette førsteutkastet sendes til veileder sammen med møteinnkalling til det aktuelle møtet. Dette gjør at veileders tilbakemeldinger og innspill kan tas til følge gjennom en eller flere revisjoner før endelig ferdigstilling av dokumentasjonen.

Dokumentasjonen har til hensikt å holde oppdragsgiver informert om fremgangen i prosjektet samt å kontinuerlig evaluere prosjektets fremdrift. Dokumentasjonen som foreligger, skal kunne etterfølges av eksterne lesere, som basert på gjeldende dokumentasjon også skal kunne implementere den løsningen vi kommer frem til i sine egne prosjekter. Det fungerer også som kontrollmekanisme slik at det som er planlagt blir gjennomført.

b. Krav til standarder og metoder

Vi tar utgangspunkt dokumentmaler utgitt av NTNU, for å sørge for tilfredsstillende utarbeidelse og kvalitet på dokumentasjonen.
De standardene vi kommer til å benytte i prosjektet er i hovedsak å lære og deretter følge såkalt «best practice» når det kommer til arbeid med Sentinel.

Når det kommer til hvilke verktøy som brukes bruker vi Microsoft 365 inkludert Teams og Sharepoint for utarbeidelse av dokumentasjon, digitale møter, samt lagring og deling av filer.

For gjennomføring av selve prosjektet vil vi benytte oss av en subscription eid av Atea, og vil få tilgang til nødvendige lisenser på etterspørsel, dette inkluderer Microsoft 365 E5 og Sentinel. Eventuelle servere som brukes i Azure vil alltid benytte de nyeste oppdateringene, funksjonalitetene og sikkerhetsoppdateringene.

c. Endringshåndtering

Endringsønsker kommer fra oppdragsgiver, brukere, prosjektdeltakerne selv og andre interessenter. Endringsønsker er vanlig og skal behandles formelt og forretningsmessig. Framgangsmåten for håndtering av endringsønsker er:

1. Dokumenter endringens innhold
2. Analyser konsekvensene for prosjektet
3. Beregn eventuell kost/nytte
4. Godkjennelse og aksept
5. Logg endringen
6. Juster planene
7. Informer interessentene
8. Gjennomfør endringen

9. Prosjektorganisering

I dette kapitlet skal vi vise hvem som er med i prosjektet og hvordan de har fordelt arbeidet mellom seg.

1. Oppdragsgiver – Atea og Morten Schjetne
2. Kvalitetskontroll – Jostein Lund
3. Prosjektgruppe – Henrik Hove Eide og Rune Sterten Marhaug.

Dette er ikke en stor prosjektorganisasjon, vi har derfor ingen formell struktur utenom de definerte rollene.

Anbefaling om videre arbeid

Ut ifra den informasjonen som ligger til grunn i denne forstudierapporten anbefaler vi at prosjektet videreføres. Prosjektet kan ta utgangspunkt i de mål og rammeverket som er utarbeidet i denne rapporten, dog med forbehold om endringer.

Prosjektet har en svært aktuell problemstilling, både for oppdragsgivers behov og prosjektgruppens mål.

10. Referanser

Forrester. (2020). *Microsoft.* Hentet fra https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RE4IgFh

Vollan, M. (2020, Januar 13). *NRK.* Hentet fra nrk.no: https://www.nrk.no/innlandet/tre-alvorlige-dataangrep-den-siste-maneden_-kripos-ser-en-okning-i-profesjonelle-hackere-1.15324245

Bacheloroppgave 10

Designrapport

Muligheter med Azure Sentinel

Et proof of concept

Av Henrik Hove Eide og Rune Sterten Marhaug

Innhold Designrapport

1. Introduksjon 39
1.1 Hensikt 39
1.2 Kort om oppdragsgiver og krav 39
1.3 Hvorfor denne løsningen 40
2. Bakgrunn 40
2.1 Avgrensing 41
2.2 Definisjoner og forkortelser 42
3. Beskrivelse av tekniske løsninger 43
3.1 Abonnement og lisenser 43
3.2 - Azure 43
3.2.1 – Resource groups 43
3.2.2 – Sentinel 44
3.2.3 – Alert rules 45
3.2.4 – Playbooks 45
3.2.5 Hunting rules 45
3.2.6 KQL 46
3.2.7 - Microsoft native connectors 46
3.2.8 Log collector 46
3.3 – Sentinel + DevOps 47
4. Detaljerte løsningsbeskrivelser 48
4.1 Forutsetninger og avhengigheter 49
4.2.1 - Azure Subscription 49
4.2.2 – Rettigheter i Azure 49
4.3 Azure PowerShell og AZSentinel 50
4.4 Krav til driftsdokumentasjon 50
4.4.1 – Forstudierapport 50
4.4.2 – Designrapport 51
4.4.3 – Driftsrapport 51
4.5 Organisatoriske og personellmessige konsekvenser 51

1. Introduksjon

1.1 Hensikt

Dette dokumentet presenterer en detaljert skisse av det ønskede sluttproduktet i dette prosjektet. Denne rapporten og produktet baserer seg på de mål, begrensinger og andre forhold som ble diskutert i forstudierapporten knyttet til dette prosjektet.

Oppdragsgiver Atea ønsker et «Proof of Concept» basert på Microsoft 365 der Azure Sentinel benyttes som SIEM og SOAR.

1.2 Kort om oppdragsgiver og krav

Oppdragsgiver i dette prosjektet er Atea. Atea er markedsleder på IT-løsninger og infrastruktur for bedriftsmarkedet i Norden og Baltikum, med mer enn 7000 ansatte og en omsetning på nærmere 40 milliarder kroner i 2020. I Norge har Atea 1600 ansatte fordelt på 22 kontorer fra Hammerfest i nord til Kristiansand i sør.

Ateas sikkerhetsgruppe ønsker at vi utvikler et system, basert på Microsoft 365 og Azure, der Azure Sentinel brukes som SIEM og SOAR. Dette betyr at systemet skal kunne automatisk oppdage trusler i tilkoblede tjenester og systemer, og kunne automatisk reagere på trusler basert på et regelsett. Som videre arbeid ønsker prosjektgruppen å automatisere utrulling, konfigurasjon og drift ved bruk av Azure DevOps. Dette er et verktøy som muliggjør det å utvikle og drifte Azure Sentinel som kode ved hjelp av PowerShell. Den andre fordelen med en slik automasjon er at en kan administrere flere kunder eller systemer fra en sentralisert kodedatabase.

Resultatet av prosjektet skal være en praktisk demonstrasjon og presentasjon av resultatene, samt fullverdig dokumentasjon av hele prosessen som vil oversendes Atea sin representant i prosjektet.

1.3 Hvorfor denne løsningen

SIEM er en moden teknologi, men er fremdeles under rask utvikling. Den neste generasjonen SIEM-verktøy, slik som Azure Sentinel, tar i bruk verktøy som Machine Learning og User Event Behavioral Analysis for å oppdage trusler. I tillegg blir SOAR en større del av SIEM-produkter. SOAR (Security Orchestrated Automatic Response) er verktøy for å automatisk respondere til trusler oppdaget av SIEM. Løsningen gir oss godt læringsutbytte og muligheten til å fordype oss i - og skalere prosjektet,slik at alle interessenter i prosjektet får det utbyttet en ønsker seg. Løsningen vil kunne raskt bli iverksatt av mulige brukere. Systemet er skalerbart og fleksibelt, og kan tilpasses brukerens behov, noe som gjør at løsningen har stor rekkevidde.

2. Bakgrunn

I sammenheng med økningen i antallet cyberangrep norske bedrifter og offentlige organisasjoner opplever, får viktigheten av IT-sikkerhet stadig mer fokus fra media og offentligheten. I denne sammenhengen blir SIEM- og SOAR-systemer ofte trukket frem som viktige deler av moderne IT-sikkerhet. Grunnen til dette er at bedrifter, som erfart av Atea, sliter med sporbarhet og visibilitet i sin IT-infrastruktur. På grunn av dette kan en bli utsatt for angrep uten å være klar over det, og en inntrenger vil ha god tid til å etablere seg i nettverket. I tillegg kan hendelseshåndteringen i etterkant ta lang tid, fordi bedriftene ikke har oversikt over dataen sin, og derfor hvilken skade som er gjort. Derfor blir SIEM tatt i bruk. SIEM er et system som samler loggdata fra alle mulige kilder i en bedrifts infrastruktur. Basert på et regelsett kan et SIEM-system automatisk oppdage uønskede hendelser i nettverket. Deretter kan en bruke et SOAR-verktøy for å automatisk respondere på hendelsen, som for eksempel å varsle personell, stenge brukerkontoer, blokkere IP-adresser eller sette maskiner i karantene.

2.1 Avgrensing

Produktet skal ikke ta hensyn til, eller dokumentere drift og oppsett av produkter tilhørende løsningen som ikke er direkte relatert til sikkerhet i Azure. Eksempler på dette er Azure AD, Intune og sikkerhetstjenester i Microsoft 365. Installasjonen av disse vil ikke bli presentert i driftsrapporten, men disse produktene og tjenestene er svært relevante for de fleste kundene til Atea, og å integrere disse produktene med Sentinel vil være en del av oppgaven.

2.2 Definisjoner og forkortelser

Active Directory Domain Services (ADDS)	Microsofts domenetjeneste som brukes til å administrere domener og brukere
Azure	Microsoft sin skyløsning
Sentinel	Microsoft sin skybaserte SIEM løsning
SIEM	Security information and event management
SOAR	Security Orchestrated Automated Response
Powershell	Scripting språk
Azure Dev Ops	Tjeneste for utvikling og deployment
Workspace	Top level ressurs for maskinlæring i Azure
Microsoft native	Tjenester og programmer utviklet av Microsoft
Log Analytics	En tjeneste i Azure som viser og behandler loggdata
Kusto Query Langauge (KQL)	Et scriptspråk utviklet for å kjøre spørringer mot log data i Azure.
IAC (Infrastructure as code)	Administrasjon av datasenter som kode
CI (Continous integration)	Nye endringer blir kontinuerlig integrert i miljøet.
CD (Continous delivery)	Korte utviklingssykluser sørger for raske oppdateringer av systemet.

CMDlets	Er kommandosnutter som brukes i powershell
UAA (User access administrator)	En rolle i Azure som trengs for å sette opp DevOps
SOC	Security Operations Centre

3. Beskrivelse av tekniske løsninger

3.1 - Abonnement og lisenser

For å ta i bruk Azure Sentinel kreves et abonnement til Azure med en tilknyttet betalingsløsning. For å gjøre Sentinel så effektivt som mulig kan det være nødvendig med abonnementer på andre tjenester, slik som Microsoft 365 E5 for å få tilgang til skysikkerhetstjenester. Dette er ikke et krav, siden Sentinel har full støtte for «on-premise» systemer.

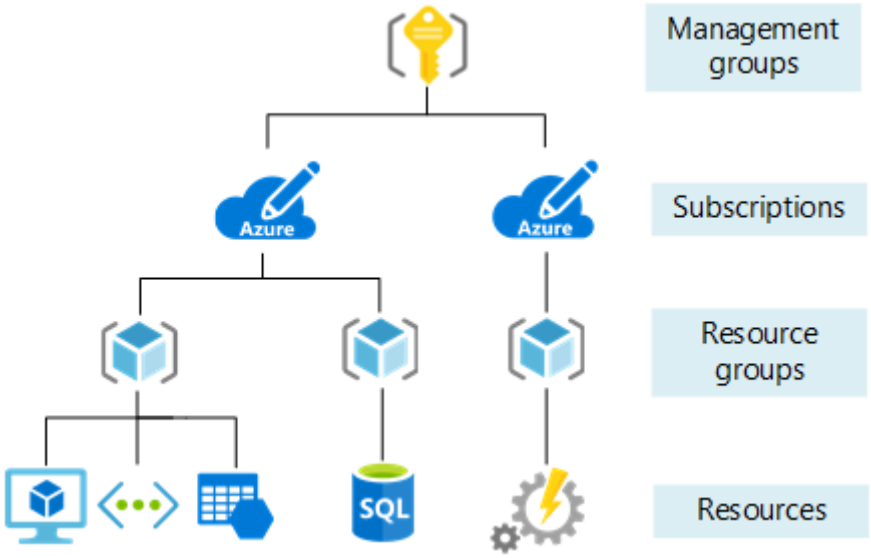
3.2 - Azure

Azure er Microsoft sin administrasjonsportal for skybaserte tjenester. Her kan du gjøre alt fra å opprette virtuelle maskiner til å sette opp din egen SIEM-løsning, du kan lage nettbutikker, servere til alle mulige formål og du sikrer deg samtidig den nyeste teknologien.

Azure er også den ledende tilbyderen av skytjenester og ble lansert i 2010. Det at de tilbyr alt av programvare, service og tjenesteytelser på internett uten at kunder trenger å bekymre seg for maskinvare, vedlikehold og andre tradisjonelle IT-utfordringer, har gjort Azure og skyplattformer veldig populære den siste tiden.

3.2.1 – Resource groups

Azure bruker følgende nivåer for ressursadministrasjon på sine tjenester. Eksempler på ressurser er virtuelle maskiner, databaser og applikasjoner.



Som figuren viser, er «management groups» det øverste nivået. Her kan man lage en gruppe som administrerer flere abonnementer, sette regler og standardisere policyer som skal gjelde for flere grupper eller abonnementer.
Under dette har en abonnementsnivået. Her kan en lage og administrere ressursgrupper. Ressursgrupper brukes til å samle et sett med ressurser som har mer eller mindre kobling til hverandre, det er en enkel måte å administrere og holde oversikt over de forskjellige ressursene man rår over på en effektiv måte.

3.2.2 – Sentinel

Sentinel er Microsoft skybaserte SIEM-løsning. Gjennom å analysere logger og data generert av tilkoblede systemer i sanntid, kan Sentinel automatisk oppdage sikkerhetstrusler. Og med videreutviklingen SOAR, kan systemet også respondere på disse truslene i sanntid. Det som gjør Sentinel et kraftig verktøy, er at det kan hente inn data fra utallige systemer, plassert på mange forskjellige steder. Videre kan det ta i bruk maskinlæring for å lære og analysere normalsituasjonen i et miljø.

Med den tekniske utviklingen vi har i dag der alt kobles sammen og alle systemer snakker med hverandre er det lett å anta at et system som Sentinel under kontinuerlig utvikling og oppdatering vil bli et viktig sikkerhetsverktøy for mange store bedrifter.

Det som også skiller Sentinel fra andre SIEM-løsninger er at det er ligger i skyen, og er derfor uendelig skalerbart. En trenger ikke gjøre store innvesteringer i hardware der en bare får

utnyttet en andel, noe som over tid sparer bedriftene for kostnader.

Det er derfor naturlig at ATEA for dette prosjektet har bestilt en SIEM-løsning basert på Sentinel.

3.2.3 – Alert rules

«Alert rules» i Sentinel brukes for å konfigurere når systemet skal varsle om at noe uvanlig eller uønsket har forekommet på en enhet eller i et miljø knyttet til Sentinel. «alert rules» kan også bli konfigurert til å kjøre en «playbook», som er en samling tiltak og motsvar til uønskede hendelser, mer om dette i neste avsnitt.

Disse reglene er designet i et tilpasset scriptspråk, Kusto Query Language eller KQL, dette er laget for at Sentinel skal kunne respondere på uønsket eller uvanlig oppførsel fra datakildene den er koblet til, det kan være alt fra automatiske rutiner til manuelle inngrep fra en SOC dersom man mistenker angrep.

3.2.4 – Playbooks

Playbooks er et sett med sikkerhetsprosedyrer som automatisk aktiveres av «alert rules», på bakgrunn av uønsket eller mistenkelig oppførsel. Der «alert rules» er egendefinerte og programmerte spørringer som skal beskytte og oppdage farer i miljøet ditt er «playbooks» laget for å respondere på ulike hendelser som oppstår.

Det som er viktig å vite med «playbooks» er at selv om det høres veldig komplisert og vanskelig ut så trenger det ikke å være tilfellet og det finnes mange gode eksempler utarbeidet av Microsoft sitt sikkerhetsteam som kan være et godt utgangspunkt, både for enkle automatiserte oppgaver og mer kompliserte oppgaver.

3.2.5 Hunting rules

Den andre typen regler en kan aktivere i Sentinel er «hunting rules». Dette er manuelt startede scripts som proaktivt jakter etter trusler. Ved bruk av hunting rules blir en med ett mer proaktiv i sikkerhetsarbeidet. Til forskjell fra «alert rules», som kjøres automatisk, er «hunting rules» bedre egnet for spørringer der det trengs et menneske for å tolke resultatene.

3.2.6 KQL

KQL også populært kalt Kusto eller Kusto Query Language er et tilpasset språk for spørringer i Azure log databaser, «monitor logs» og «Azure Monitor Application Insights». KQL er nært beslektet SQL, men kan ikke brukes til å endre på databaser. Det er altså i dette språket en formulerer spørringer for å søke i og forstå logger og hendelser i systemet en administrerer.

3.2.7 - Microsoft native connectors

«Microsoft native connectors» brukes for å koble Microsoft sine programmer og tjenester til Sentinel og konfigurerer disse til å kunne kommunisere sammen og utveksle data, det finnes nå over 90 prekonfigurerte koblinger i Sentinel, med enkle oppsett og guider for å enkelt og effektivt få koblet datakildene våre til Sentinel.

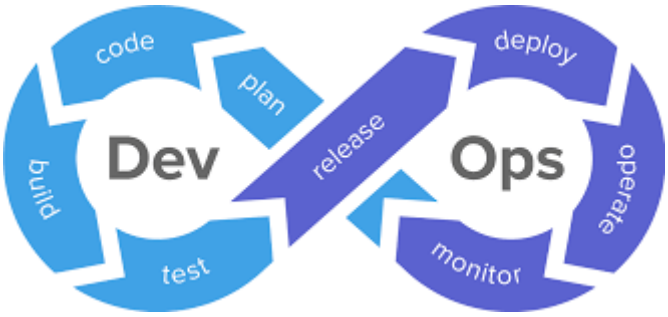
3.2.8 Log collector

En «log collector» er en maskin som brukes for å samle inn og standardisere formatet på logger som motas fra enheter som ikke har direkte tilkobling til Azure Sentinel, altså ligger de ikke i Azure. En slik log collector er en maskin som står mellom Sentinel og miljøet, og sikrer at dataene som kommer inn fra disse datakildene blir standardisert i et universelt format og verifisert av Sentinel.

Siden Sentinel er såpass nytt er det unaturlig å tenke at de dekker alle mulige leverandører og konfigurasjoner av all programvare, derfor kan en slik «log collector» være nyttig siden den muliggjør at alle enheter kan integreres med Sentinel.

«Log collectoren» er en Linux basert maskin som må ha minimum 4 CPU kjerner og 16GB minne for å kjøre optimalt, når den er påslått og riktig konfigurert vil den motta loggdata fra andre enheter i miljøet, og videresende disse til Sentinel. Vi kommer mer inn på oppsett og konfigurasjon av «log collectors» i driftsrapporten for dette prosjektet.

3.3 – Sentinel + DevOps

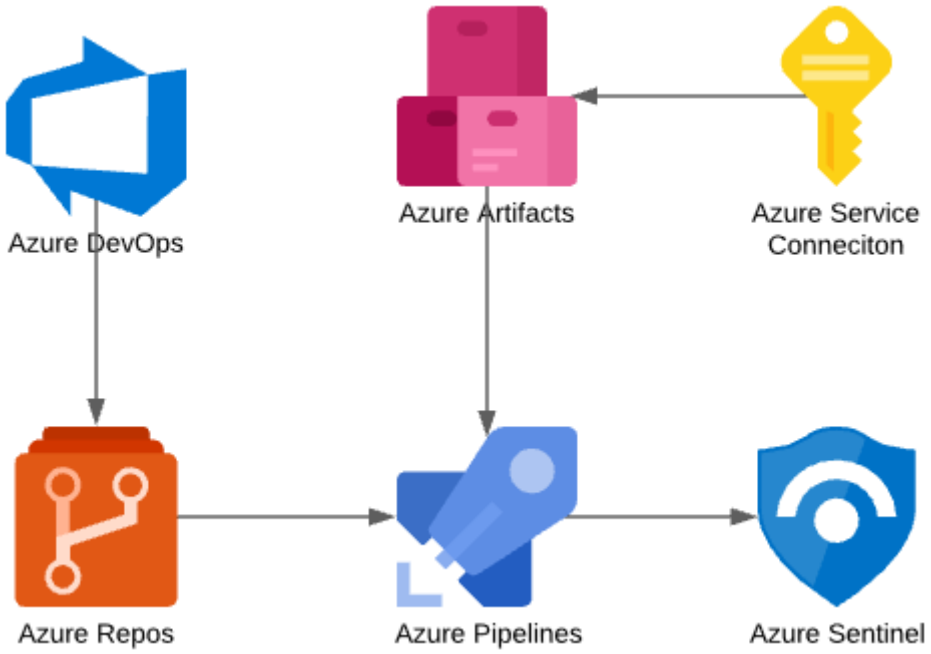


DevOps er en arbeidsform og et system som kombinerer utvikling (Dev) med drift (Ops). Et slikt system er nesten nødvendig å ha på plass for å kunne dra full nytte av Sentinel. Sentinel er et system som krever kontinuerlig utvikling for å kunne møte nye trusler fra omverdenen. Derfor passer DevOps utmerket til å arbeide med Sentinel, fordi det tillater prosjektgruppen til å ha korte utviklingssykluser og kontinuerlige oppdateringer.

I dette prosjektet er det bestemt å bruke DevOps som et verktøy for å automatisere og effektivisere utrulling og konfigurasjon av Sentinel. Denne typen arbeid havner under uttrykket «Infrastructure as Code» (IaC). Dette er en prosess som innebærer å administrere IT-infrastruktur, basert på konfigurasjonsfiler som kan leses av maskiner fremfor å konfigurere maskiner og systemer manuelt. Med DevOps kan vi installere, konfigurere, administrere og utvikle Sentinel på flere separate abonnenter og tenants, alt fra samme kodebase.

Azure DevOps er en tjeneste som tilbyr alt som trengs for denne typen arbeid. I Azure DevOps kan en opprette team, planlegge sprints og målsetninger og administrere en kodebase. For dette prosjektet vil vi kun ta i bruk tjenestene Repos, Testing og Pipelines.

Repos er et helt vanlig «git repository», og blir brukt for å lagre kode og føre versjonskontroll. Det er her Pipelines henter scripts fra. Pipelines brukes for å dytte endringer gjort i koden til Azure Sentinel.



Virkemåten er som følger: En pipeline blir aktivert for å oppdatere Sentinel når det blir gjort endringer i koden. Informasjon om Azure og Sentinel blir hentet fra en Resource Manager, dette er en tjeneste som henter informasjon fra Azure. Denne informasjonen blir lagret i et artefakt, en fil som kan brukes av flere scripts senere i pipelinen. Deretter startes en rekke PowerShell scripts for å validere konfigurasjonsfilene, kobler seg opp til Azure med variabler fra artefakten, og oppdaterer Sentinel med de nye endringene i koden.

4. Detaljerte løsningsbeskrivelser

Her finnes en detaljert beskrivelse av hvilke komponenter systemet består av, hva som skal til for at de skal fungere, det står også litt om hvilke verktøy vi har brukt for å løse prosjektet.

4.1 Forutsetninger og avhengigheter

4.2.1 - Azure Subscription

En aktiv Azure Subscription er nødvendig for å kunne drifte og kjøre de operasjonene som trengs for å drifte Sentinel og de nødvendige tjenestene for innhenting og analyse av logger og annet datamateriale.

4.2.2 – Rettigheter i Azure

Før en starter arbeidet med Sentinel må en ha tilstrekkelige rettigheter til å utføre handlingene som kreves for å sette opp Sentinel. Uten tilstrekkelige rettigheter vil man ikke kunne få gjennomført prosjektet, og oppsettet vil før eller siden feile.

Rettigheter kan enkelt sjekkes ved å gå til Subscriptions i Azure portalen og se under «My permissions», her vil det stå hvilket rettighetsnivå den gjeldende brukerkontoen har og hvilke eventuelle rettighetsgrupper brukeren tilhører.

For at Azure Sentinel skal fungere krever det at det kommer data inn, Dette gjøres av agenter agenter er små miniprogrammer som er programmert til å gjennomføre en spesifikk oppgave, eller gjøre en konfigurasjon på vegne av den som deployer. På native Microsoft tjenester og programmer rapporterer direkte til Microsoft Sentinel så lenge enheten er koblet til workspacet. Et workspace i azure er en top-level resource som samler alle artifaktene du bruker å jobbe med når du bruker maskinlæring. Det logger alle tester, metrics og outputs. Den lagrer også snapshots av script man prøver å kjøre.

Følgende rettigheter kreves for å sette opp Sentinel:

- «User access administrator» rettigheter i Azure er en rettighet som trengs for å koble DevOps sammen med Azure. Dersom man ikke har denne rettigheten, vil ikke DevOps pipelines kunne hente informasjon fra Azure eller dytte endringer til Sentinel.
- Contributor rollen i Azure gir full tilgang til å styre alle ressurser i Azure, men gir ikke tilgang til brukeradministrasjon.

4.3 Azure PowerShell og AZSentinel

Azure PowerShell er en samling cmdlets for PowerShell, det vil si en samling av kommandoer i PowerShell som kan brukes for å administrere mange tjenester i Azure. Dessverre finnes det enda ingen cmdlets for å administrere Sentinel. Derfor har det blitt laget en utvidelse til PowerShell som heter AzSentinel. Denne utvidelsen til PowerShell muliggjør det å administrere Sentinel via API. Denne kan installeres fra PowerShell Gallery med følgende kommando:

```
Install-Module AzSentinel -Scope CurrentUser -Force
```

4.4 Krav til driftsdokumentasjon

I løpet av dette prosjektet skal det utredes følgende dokumentasjon for å dokumentere alle aspekter ved prosjektet når det er under utvikling, alt fra planleggingsfasen til gjennomføringen skal dokumenteres og det skal resultere i følgende dokumentasjon.

4.4.1 – Forstudierapport

Forstudierapporten skal dokumentere hvilke behov man trenger og gi grunnlag for hvorfor et prosjekt skal gjennomføres, det skal dokumenteres hvem som er parter i prosjektet, hvilke behov oppdragsgiver har og hvem de ønsker skal løse oppgaven. Det skal også utarbeides en plan for når prosjektet skal starte, og når prosjektet skal leveres.

Det skal også utarbeides mål og rammeverk for prosjektet. Målene skal si noe om hva man ønsker å oppnå og hvordan man ønsker å komme dit, siden skal man også sette økonomiske og andre relevante begrensninger for prosjektet slik at alle parter vet hva de skal forholde seg til og som sørger for klarhet i kommunikasjon og en mer formalisert avtale mellom oppdragsgiver og oppdragstaker.

4.4.2 – Designrapport

Designrapporten skal inneholde avgrensninger, forklare hvilke tekniske løsninger man benytter seg av for å løse prosjektet samt hvordan disse blir brukt for å løse det gitte prosjektet konkret.

Den skal også gi oversikt over hvilke krav man stiller til prosjektet, hardware, software, lisensiering, tjenester og eventuelt personell og organisatoriske endringer som kreves for å løse prosjektet på forespeilet måte. Den skal også inneholde en kort redgjørelse for dokumentasjonen som blir utarbeidet gjennom prosjektet.

4.4.3 – Driftsrapport

Driftsrapporten er en rapport som skal vise hvordan man har utviklet de forskjellige løsningene man bruker i prosjektet, hvordan de er konfigurert og satt opp og hvordan de hører sammen. Det skal fungere som en steg for steg installasjonsguide slik at hvem som helst skal kunne lese driftsrapporten, kopiere stegene som blir gjennomgått og oppnå samme resultat som forfatterne av driftsrapporten.

Driftsrapporten skal også fungere som oppslagsverk for oppsett og konfigurasjon de forskjellige komponentene som blir benyttet i prosjektet. prosjektet.

4.5 Organisatoriske og personellmessige konsekvenser

SIEM-løsninger har begrensinger som gjør dem ineffektive dersom den riktige støttestrukturen fra personell og tredjeparts programvare er på plass. SIEM kan ikke erstatte brannmurer og IDS (Intrusion Detection Systems), men innehar en supplementerende rolle i en organisasjons IT-sikkerhet, fordi det gjør det enklere å oppdage trusler i systemene som er tilkoblet. Et SIEM-verktøy må tilpasses den enkelte bedriftens trusselbilde, og kontinuerlig overvåkes og forbedres for å møte nye trusler. Dette betyr at å ta i bruk et SIEM-verktøy kan være ressurskrevende, både personellmessig og økonomisk. Det er også her organisasjoner må ta stilling til om de skal gå et skritt videre og integrere SOAR, det vil si automatisk håndtering av hendelser oppdaget av SIEM-verktøyet, slik at det ikke nødvendigvis kreves personell til 24/7 overvåkning av systemet

4.6 Milepæler og datoer

Milepæler	
Hendelse	Dato
Oppstart av prosjektet	11.01.2021
Ferdigstillelse av forstudierapport	31.01.2021
Ferdigstillelse av designrapport	21.03.2021
Ferdigstillelse av driftsrapport	08.05.2021
Ferdigstillelse av sluttrapport	19.05.2021
Innlevering av dokumentasjon	20.05.2021
Presentasjon av prosjektet	28.05.2021

Driftsrapport

Muligheter med Azure Sentinel

Et proof of concept

Henrik Hove Eide og Rune Sterten Marhaug

Revisjonslogg

Dato	Versjon	Beskrivelse	Forfatter
18.05.2021	1.1	Rettskriving og siste gjennomgang før levering	Henrik

Forord

Denne rapporten er utarbeidet på oppdrag fra Atea, og har til hensikt å være både brukerveiledning og installasjonsveileder for systemet som er utarbeidet i et bachelorprosjekt. Prosjektets mål er å bygge et «proof-of-concept» basert på Microsoft 365, der Azure Sentinel brukes som SIEM-løsning. Vårt fokus for arbeidet, og derav denne rapporten, har vært å gjøre implementeringen og driften av Sentinel i stor skala så lettvint som mulig. Dokumentet gir en gjennomgående innføring i de delene av Sentinel som er tatt i bruk i systemet, og viser detaljert hvordan prosjektet vårt kan etterfølges.

Rapporten fokuserer på hvordan Sentinel, sammen med Azure DevOps, etableres i et Azure-miljø. Vi demonstrerer hvordan en kan skrive og aktivere regler for deteksjon og hendelseshåndtering (playbooks, analytics- og hunting rules), både manuelt i Azure portalen og gjennom DevOps. Dokumentet følger prosjektets gang i kronologisk rekkefølge. Det er likevel mulig å hoppe i dokumentet dersom leseren ønsker informasjon om bestemte deler av systemet. Den grunnleggende konfigurasjonen av systemet blir gjennomgått i kapittel 3, dette kapittelet bør være gjennomgått før en kan plukke individuelle deler fra kapittel 4.

Prosjektgruppen består av Henrik Hove Eide og Rune Sterten Marhaug, studenter ved NTNU. Med oss som veiledere har vi Jostein Lund fra NTNU og Morten Schjetne fra Atea.

Innhold

Revisjonslogg 54

Forord 54

1. Om dette dokumentet 57

1.1 Hensikten med dokumentet 57

1.2 Avgrensning 58

1.3 Dokumentets oppbygging 59

2. Prosjektets overordnede struktur 60

3. Oppstart 61

3.1 Grunnleggende konfigurasjon av Azure miljø 61

Nødvendige roller og lisenser 61

Opprette Log Analytics Workspace 62

Aktivere AzureCLI 63

3.2 Grunnleggende konfigurasjon av Azure DevOps 64

3.2.1 Opprette organisasjon og prosjekt i Azure DevOps 66

3.2.2 Oppsett av git og repos 68

3.2.3 Opprette en Resource Manager 70

Opprette variabelgruppe 72

3.2.4 Klargjøring av pipelines 75

Oppsett av scripts pipeline 76

Aktivering av Sentinel 80

Aktivering av connectors 81

Oppsett av connectors pipeline 83

4. Utførelse 84

4.2 Oppsett av Log Collector til Azure Sentinel 84

4.3 Aktivere connectors 89
4.3.1 Microsoft native connectors 89
4.4 – KQL 95
4.5 Analytic rules 96
4.5.1 Analytic rules i Sentinel 96
4.5.2 Deployment av rules fra DevOps 103
4.6 Skrive og aktivere playbooks 104
4.6.1 Playbooks i Sentinel 104
4.6.2 Playbooks i DevOps 109
4.7 Skrive og aktivere hunting rules 110
4.7.1 Hunting rules i Sentinel 110
4.7.2 Hunting rules i DevOps 111
4.8 Fremtidig arbeid 112
Referanser 113

Forkortelser og definisjoner	
Agents	Små programmer som utfører sin oppgave når de kjøres
EPS	Events per sekund
Primærnøkkel	Primærnøkkelen til workspacet i Azure Sentinel
VM	Virtuell maskin
WS_ID	Workspace ID i Azure Sentinel
Azure RM	Azure Resource Manager
CEF-logger	Common Event Format

1. Om dette dokumentet

1.1 Hensikten med dokumentet

Denne rapporten er skrevet i forbindelse med en bacheloroppgave gjennomført av Henrik Hove Eide og Rune Sterten Marhaug, i samarbeid med Atea. En driftsrapport har til hensikt å beskrive i detalj hvordan løsningen vi har kommet frem til er bygd opp. Leseren skal kunne hente ut forståelse for systemets oppbygning og kan enkelt etterfølge prosjektet.

Driftsrapporten er bygd på utgangspunktet presentert i designrapporten, likevel vil det ferdige systemet ha en del forskjeller fra det som er presentert i designrapporten. Dette kommer av at vi underveis har lært mer om systemet vi har bygd, og funnet bedre måter å løse oppgaver på. Vi har ikke gått tilbake for å endre på designrapporten i stor grad, heller vil større endringer på systemet bli diskutert i sluttrapporten.

1.2 Avgrensning

Systemet som blir presentert i denne rapporten baserer seg i utgangspunktet på målene beskrevet i forstudierapporten, mens utførelsen baserer seg på designrapporten, som beskriver hvilke løsninger som er nødvendige for å oppnå målene i prosjektet. Det har oppstått endringer i prosjektets fokus underveis, men dette blir diskutert i sluttrapporten. Her presenterer vi systemet slik det har blitt.

Denne rapporten presenterer den endelige løsningen med følgende egenskaper utarbeidet i forstudiet:

- Systemet kan automatisk oppdage trusler i Microsoft 365-produkter.
- Systemet kan automatisk respondere på trusler.
- Systemet er skalerbart.

Dette dokumentet avgrenses til et «proof-of-concept» der funksjonalitet i Azure Sentinel blir prøvd mot Microsoft 365 i Microsoft Azure.

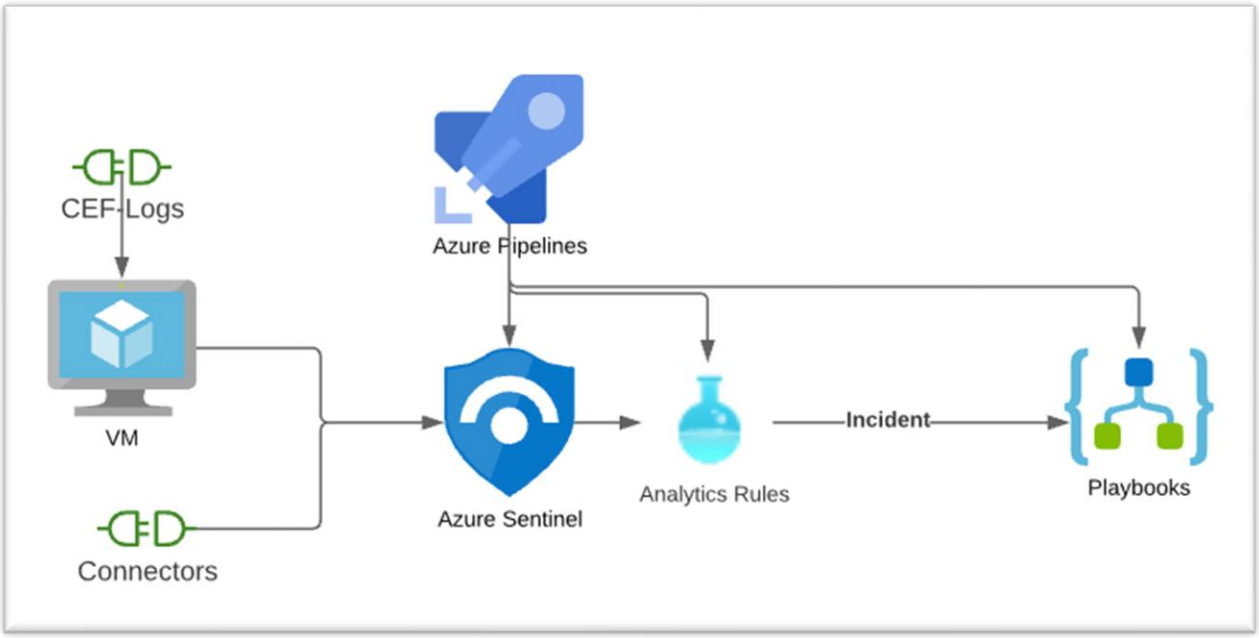
1.3 Dokumentets oppbygging

Dette prosjektet er omfattende. Derfor har vi for å sikre leseren en god forståelse av systemet, valgt å forklare dokumentets oppbygging nærmere her. Denne rapporten er primært skrevet med tanke på at leseren skal følge utarbeidelsen fra start til slutt, grunnet kompleksiteten i systemet er det mange gjensidige avhengigheter. Derfor er rapporten todelt. I kapittel 3, «oppstart», går vi gjennom grunnleggende funksjonalitet og konfigurasjon av systemet, samtidig som vi forklarer valgene som er gjort. Grunnen til dette er at leseren kan etter å ha fulgt kapittel 3, velge selv fra kapittel 4, «utførelse», hvilke deler av systemet hen ønsker å ta i bruk. Vi anbefaler selvsagt at rapporten likevel følges fra start til slutt.

I starten av hvert kapittel vil vi forsøke å bygge en rask forståelse av hver enkelt del, og raskt forklare hvorfor vi har valgt å ta det i bruk. Deretter følger en installasjonsveiledning som viser hvordan vi har satt opp hver enkelt del av systemet.

Vedlagt denne rapporten er en zip-mappe med konfigurasjonsfiler og skript som er nødvendige å ta i bruk for de delene av rapporten som handler om Azure DevOps. Dessverre kan vi ikke offentlig dele prosjektet vårt i DevOps. Grunnen til dette forklarer vi i rapporten der det er relevant. Skulle leseren derimot ønske tilgang, for å klone eller få innsikt i repoet, er det bare å ta kontakt. Til tross for dette bør dokumentasjonen i dette dokumentet være grundig nok til at prosjektet kan etterfølges uten tilgang til filene.

2. Prosjektets overordnede struktur



Sammenhengen i Sentinel er som følger. Sentinel mottar loggdata fra både connectors som enkelt kan aktiveres i Sentinel og CEF-logger fra en log collector VM. Denne informasjonen blir så sammenlignet med analytics rules. Det er ulike typer regler, som vi forklarer senere i rapporten. Reglene kommer fra tre kilder, via DevOps, opprettet direkte i portalen eller forhåndsdefinerte regler fra Microsoft. Basert på regelsettet vil det til slutt genereres «incidents», dette er hendelser av interesse som enten blir behandlet automatisk av en playbook, eller må manuelt håndteres av en person. I tillegg har vi hunting rules, som lar en bruker proaktivt jakte etter trusler ved å skrive spørringer mot loggdatabasen. Den siste funksjonen i Sentinel er Workbooks, som gjør det enkelt å presentere, og få oversikt over, informasjonen som blir generert av Sentinel.

3. Oppstart

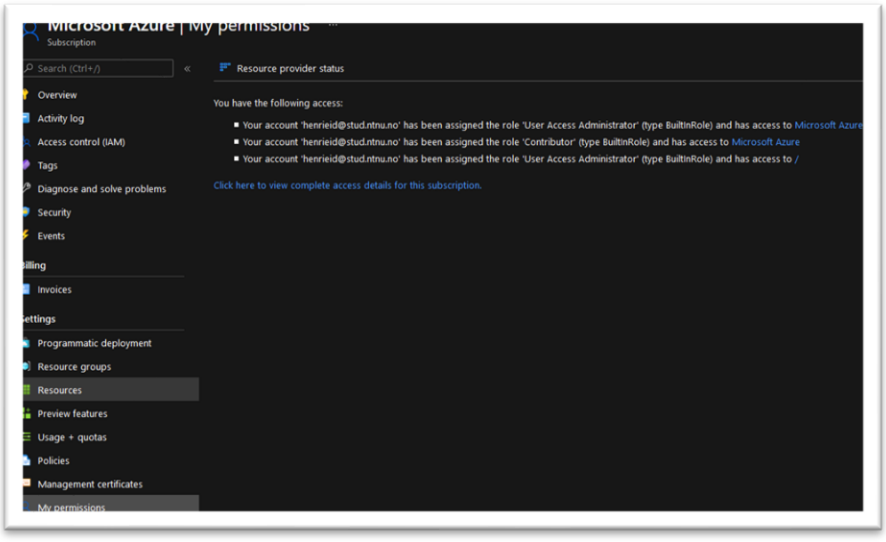
Oppstarten av arbeidet i prosjektet handlet i stor grad om å gjøre seg kjent med Azure portalen, Sentinel og relaterte produkter. Vi antar at leseren er kjent med Azure portalen. Dersom leser skulle være usikker, refererer vi til beskrivelse av tekniske løsninger i designrapporten for en grundigere innføring i den generelle strukturen til Azure, DevOps og Sentinel.

Dette kapittelet har oppstått parallelt med arbeidet med prosjektet, og har blitt endret etter hvert som vi oppdaget nye avhengigheter og funksjoner vi ønsket å ta i bruk. Her beskriver vi den grunnleggende konfigurasjonen som må gjøres i henholdsvis i Azure og Azure DevOps for videre arbeid. Spesielt gjelder dette rettigheter og å sørge for at DevOps og Sentinel kan kommunisere.

3.1 Grunnleggende konfigurasjon av Azure miljø

3.1.1 Nødvendige roller og lisenser

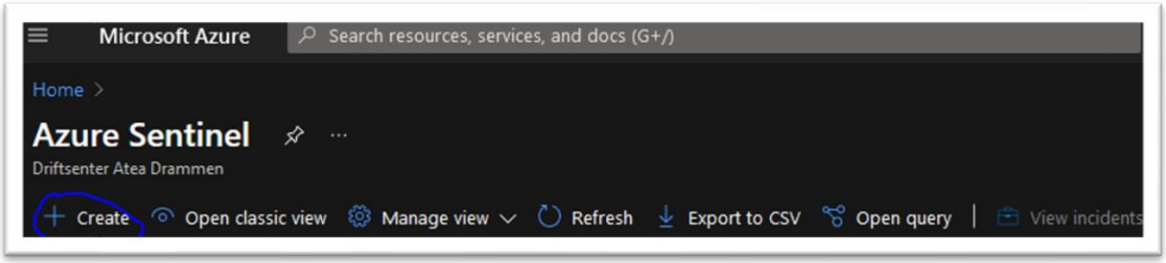
For dette prosjektet trengs en brukerkonto i Azure med rollene «User Access Administrator» og «Contributor». Roller kan sjekkes ved å gå til Subscription -> Subscription-navn -> My Permissions.



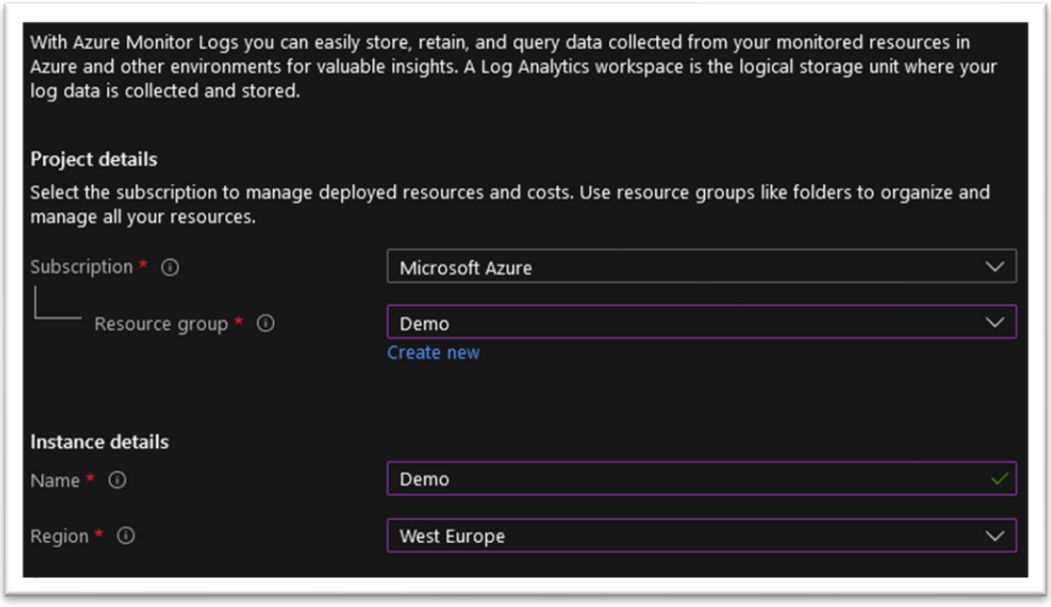
Vi disponerer en subscription til Azure som er administrert av Atea, vi har derfor ikke innsyn i kostnadene og produktene Atea har i sin avtale. Den eneste lisensen vi har spesielt bedt om er en Azure AD Premium P2 lisens som blir brukt for å kunne sende sign-in logs fra Azure AD til Sentinel. Dette er ikke nødvendig for å gjennomføre prosjektet, men det er verdt å huske at virkningsgraden til Sentinel skalerer med mengden data den har tilgengelig.

Opprette Log Analytics Workspace

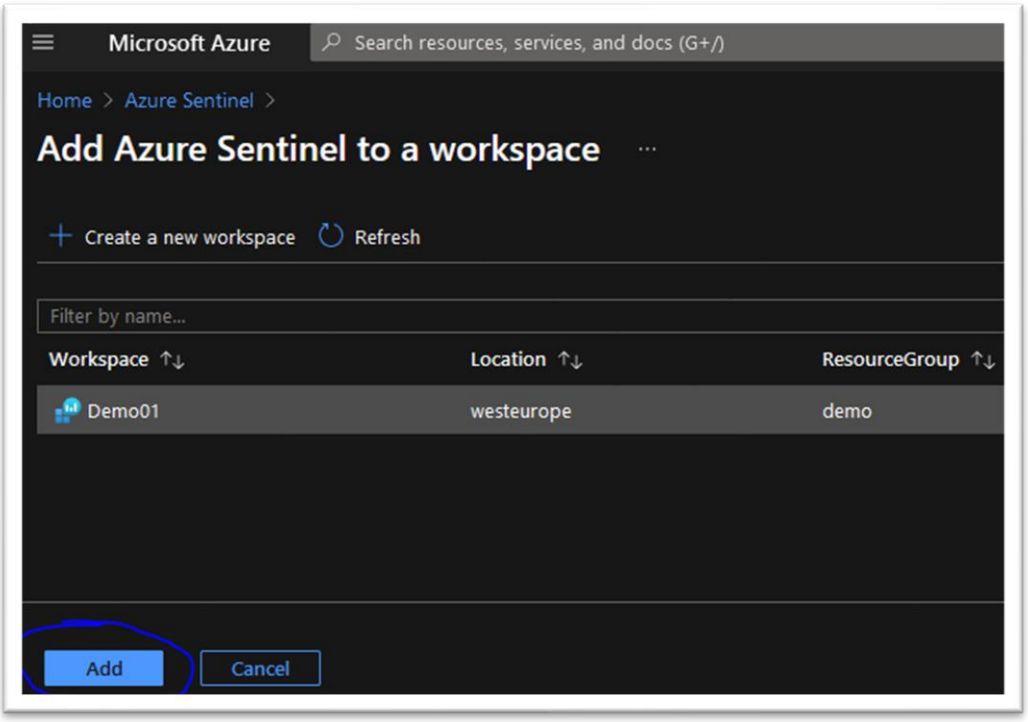
Et Log Analytics Workspace er en container for loggdata, og er den grunnleggende enheten for å lagre og organisere data i Sentinel. Alle connectors vil sende data via Log Analytics.



Trykk på «create» og så «create a new workspace».



Fyll inn skjemaet og opprett en Resource Group dersom du trenger det.



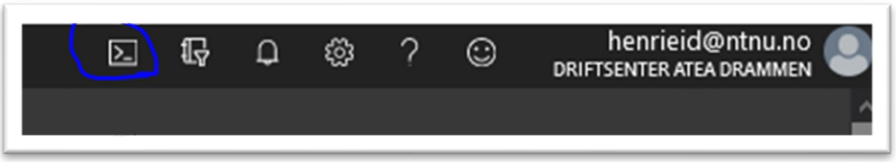
Klikk på den nyopprettede workspacet og trykk add for å legge det til Sentinel. Skriv ned navnet på workspacet og ressursgruppen den ligger i. Det vil vi få bruk for senere.

Aktivere AzureCLI

AzSentinel er en PowerShell-modul laget av Wortell[3], modulen har en rekke script som gjør det enkelt å administrere Sentinel, spesielt gjør den det enkelt å bruke Sentinel sin API for automasjon. Dette må gjøres fordi ikke alle delene av Sentinel enda støtter Azure Resource Manager (ARM). Vi kommer tilbake til dette senere.

For å ta i bruk denne modulen har vi valgt å bruke Azure CLI. Dette er et program som kan installeres lokalt, eller brukes direkte i nettleseren. Å bruke Azure CLI i nettleseren vil påløpe ekstra kostnader for en storage account. Azure CLI i nettleseren finner du i menyen øverst.

[3] https://github.com/wortell/AZSentinel



Så installerer du modulen med følgende kommando:

```
PS /home/henrik> Install-Module AzSentinel -Scope CurrentUser -Force
```

Ved å bruke denne modulen kan vi blant annet hente ut alert- og hunting rules, playbooks og workbooks fra Azure. Fordelen med dette er at en kan i første omgang opprette regler manuelt i Azure portalen, dette er enklere enn å skrive konfigurasjonsfilene direkte. I tillegg er det enkelt å verifisere at reglene fungerer som de skal. Når en henter ut reglene kan en være sikker på at de fungerer som de skal. Vi kommer tilbake til hvordan vi konkret bruker denne Powershell-modulen senere i rapporten.

3.2 Grunnleggende konfigurasjon av Azure DevOps

Et av hovedfokusene for prosjektet er å automatisere Azure Sentinel i størst mulig grad, dette gjøres ved bruk av et verktøy som heter Azure DevOps. DevOps lar oss ta i bruk prinsippene CI/CD (Continous Integration og Continous Delivery) for å drifte Sentinel som kode. Fordelene med dette er mangfoldige, blant disse er muligheten for kildekontroll, som gjør det mulig å spore endringer i Sentinel, og ikke minst at en kan med kodebasen som utgangspunkt rulle ut endringer til flere tenants samtidig.

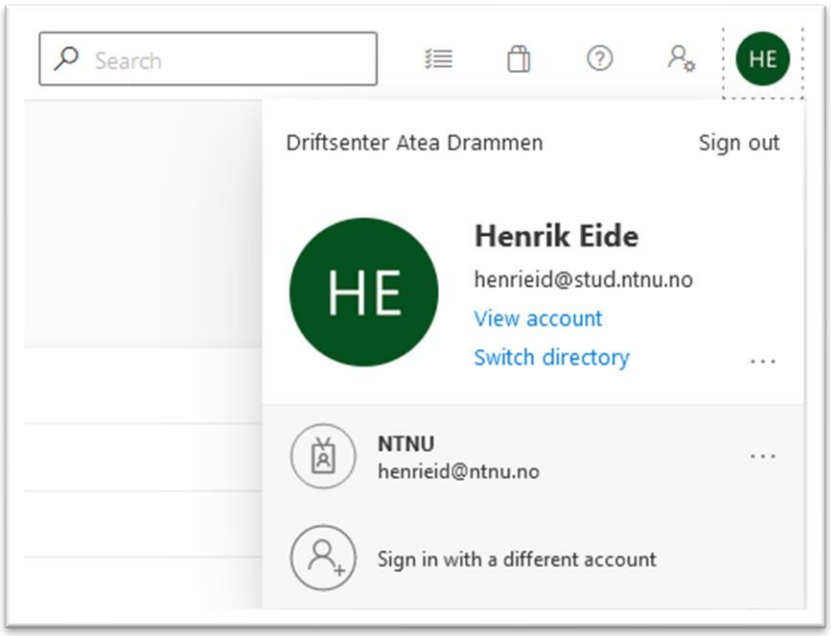
Siden det ikke enda finnes støtte for bruken av ARM (Azure Resource Manager) for alle deler av Sentinel, må vi bruke andre metoder for å automatisere ulike komponenter av Azure Sentinel. Tabellen under viser hvilke komponenter vi kan automatisere, og hvilke verktøy vi kan bruke for å oppnå dette.

Komponent	Automatisert med
Alert rules	API, Powershell, ARM
Onboarding	API, Powershell
Hunting rules	API, Powershell
Playbooks	ARM
Workbooks	ARM
Connectors	API

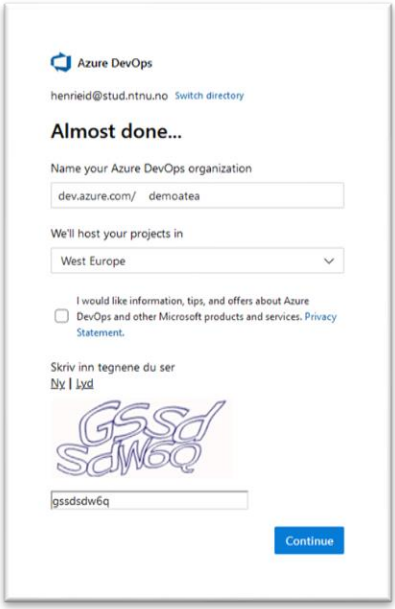
- **Powershell:** Vi tar i bruk en modul for Powershell som heter AzSentinel, som gjør det enkelt å automatisere de komponentene som støttes av denne modulen.
- **API:** Sentinel har en offentlig API, denne brukes kun for å aktivere connectors, siden de andre komponentene kan automatiseres med Powershell eller ARM.
- **ARM:** Azures innebygde platform for infrastructure as code (IaC), støtter management og deployment av Azure ressurser. Selve filformatet er json.

3.2.1 Opprette organisasjon og prosjekt i Azure DevOps

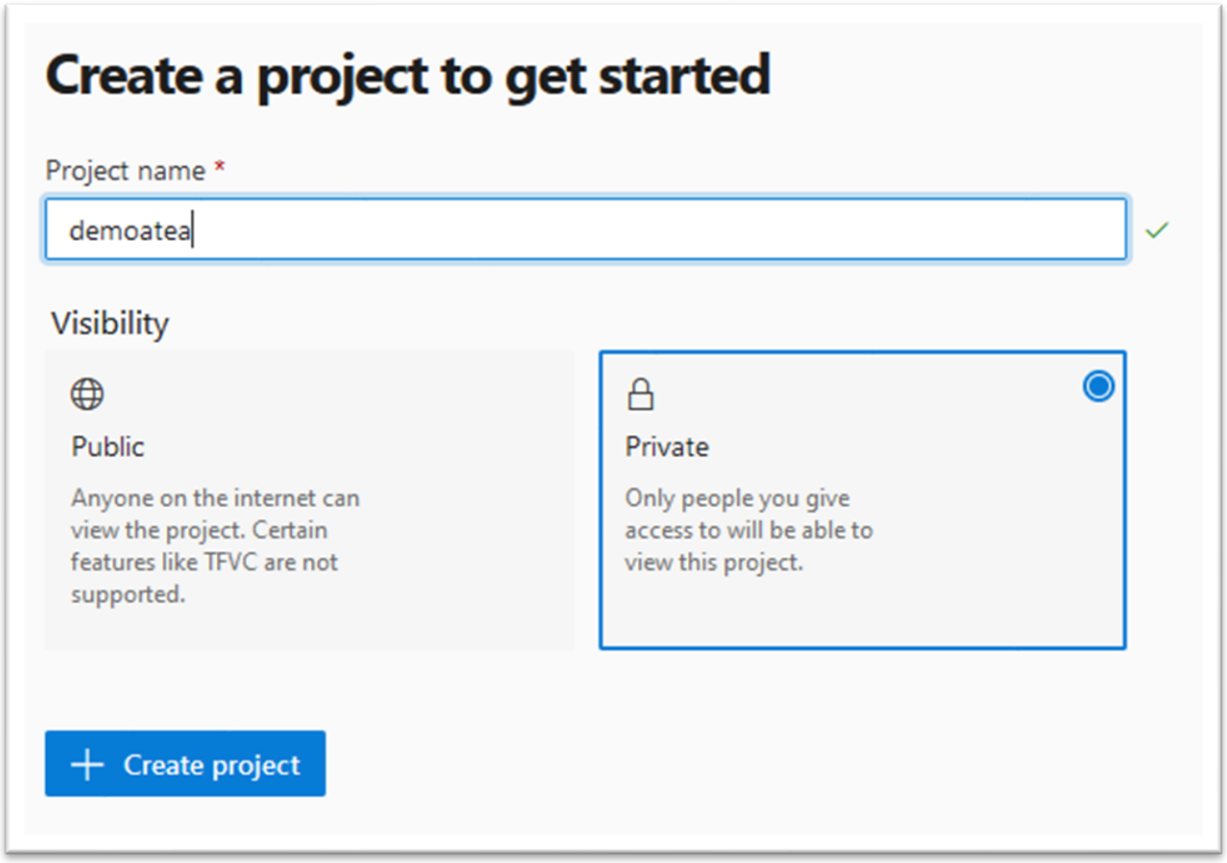
Viktig: Før du går videre, sørg for at du arbeider i riktig directory i DevOps. Dette sjekker du enkelt ved å trykke på navnet ditt øverst til høyre. Dette er fordi vi i dette prosjektet autentiserer DevOps pipelines med din gjeldende Azure log-in.



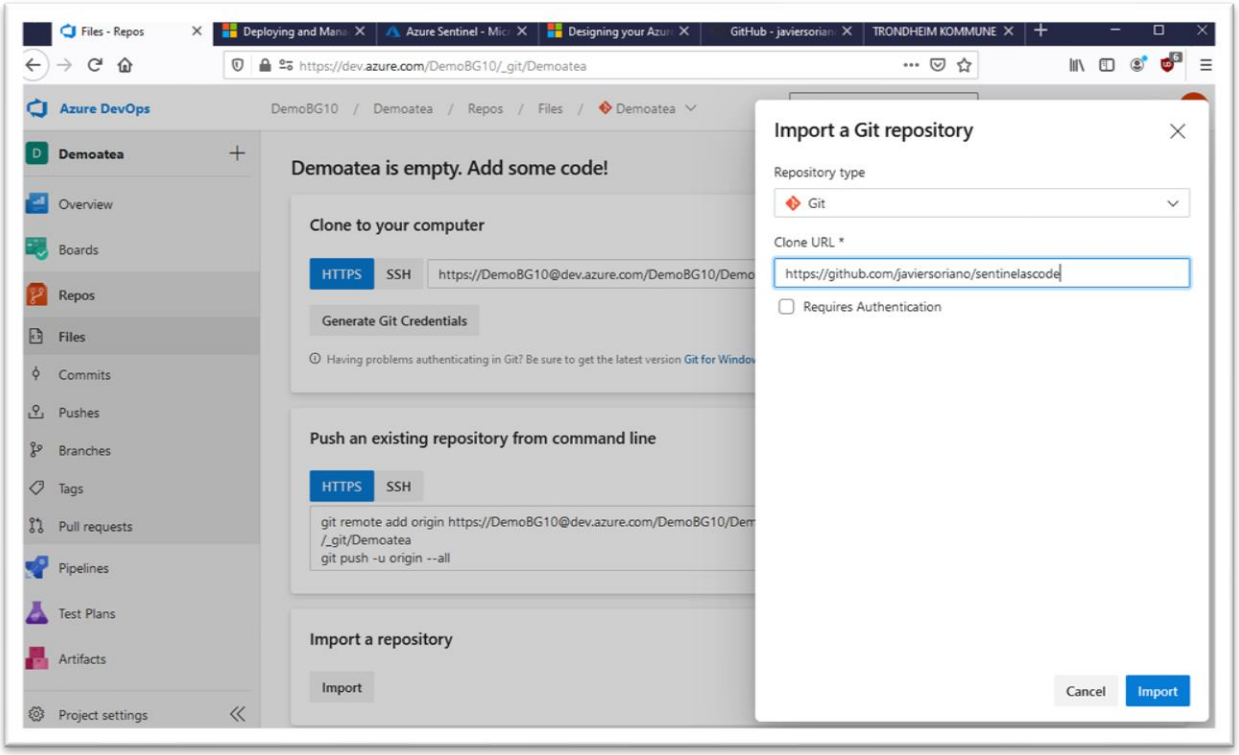
Gå til dev.azure.com og logg inn. I sidepanelet trykker du «New organization». Gi organisasjonen et navn, og sørg for å velge samme region som Sentinel, siden det kommer ekstra kostnader knyttet til å flytte data mellom ulike datasenter.



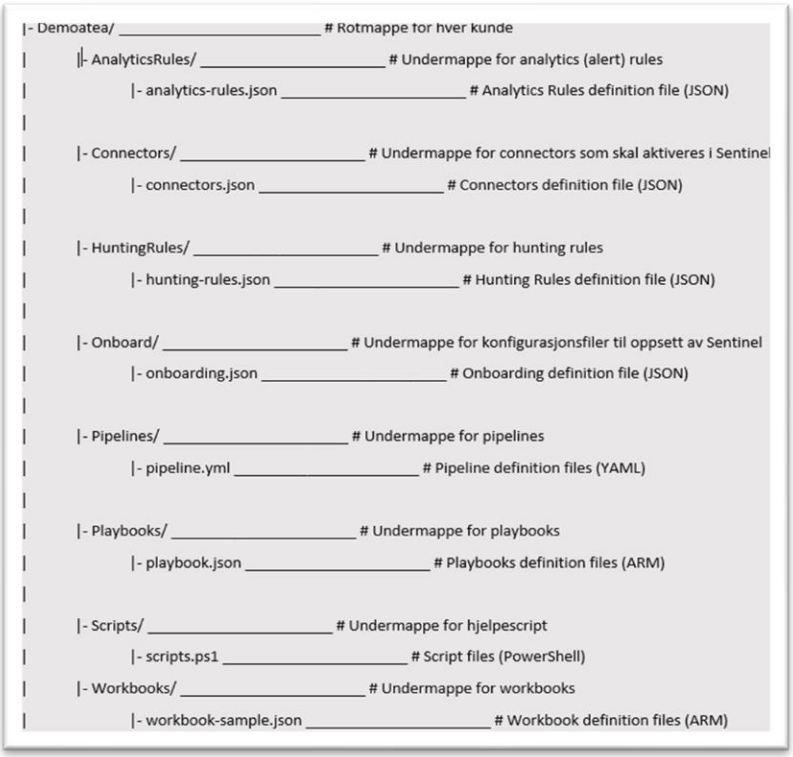
Når organisasjonen er opprettet vil du ha muligheten til å opprette ditt første prosjekt. Gi prosjektet et passende navn og sett det som privat, dette er fordi «public projects» har et noen begrensninger som er kritiske for dette prosjektet.


Create a project to get started
Project name *
demoatea
Visibility
Public
Anyone on the internet can view the project. Certain features like TFVC are not supported.
Private
Only people you give access to will be able to view this project.
Create project

3.2.2 Oppsett av git og repos



Under repos-menyen importerer vi et github repo som vi skal arbeide videre med. Trykk på import og last opp filene som er vedlagt denne rapporten.

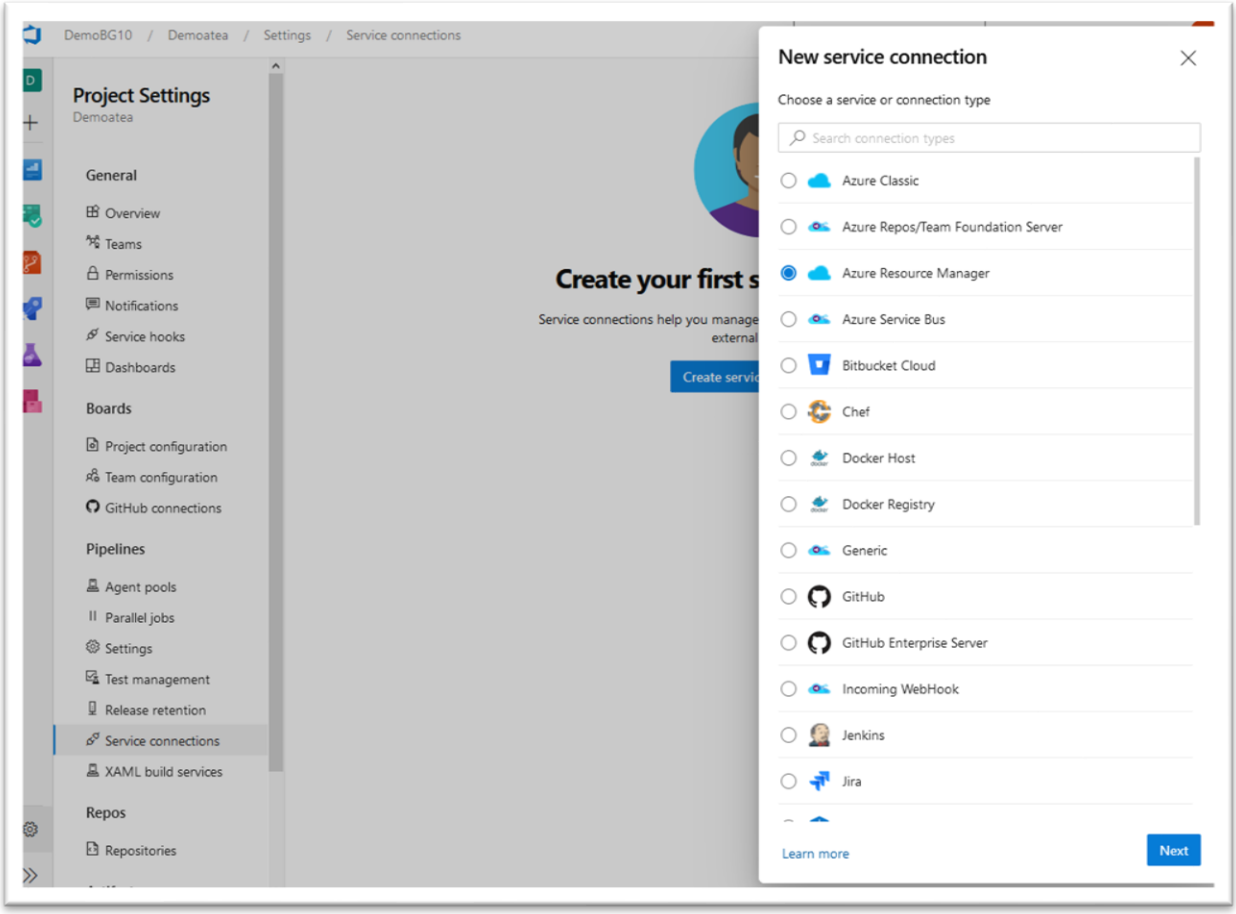


Her er en oversikt over strukturen i repoet. Hver mappe representerer de ulike funksjonene i Sentinel som kan automatiseres med pipelines.

Connectors	Starter automatisk data sources i Sentinel. Dette fungerer kun med kilder som allerede er støttet av Sentinel, slik som M365.
Analytic Rules	Alle analytics rules som skal tas i bruk i Sentinel
Hunting Rules	Alle hunting rules som skal tas i bruk i Sentinel
Onboard	Aktivering av Sentinel i definerte workspaces
Pipelines	YAML-filer som definerer CI/CD pipelines for automasjon av Sentinel deployments
Playbooks	Playbooks i Sentinel, også kjent som automation rules
Workbooks	Workbooks
Scripts	Powershell script som blir brukt for å bla. validere kode og automatisere onboarding, alert- og hunting rules.

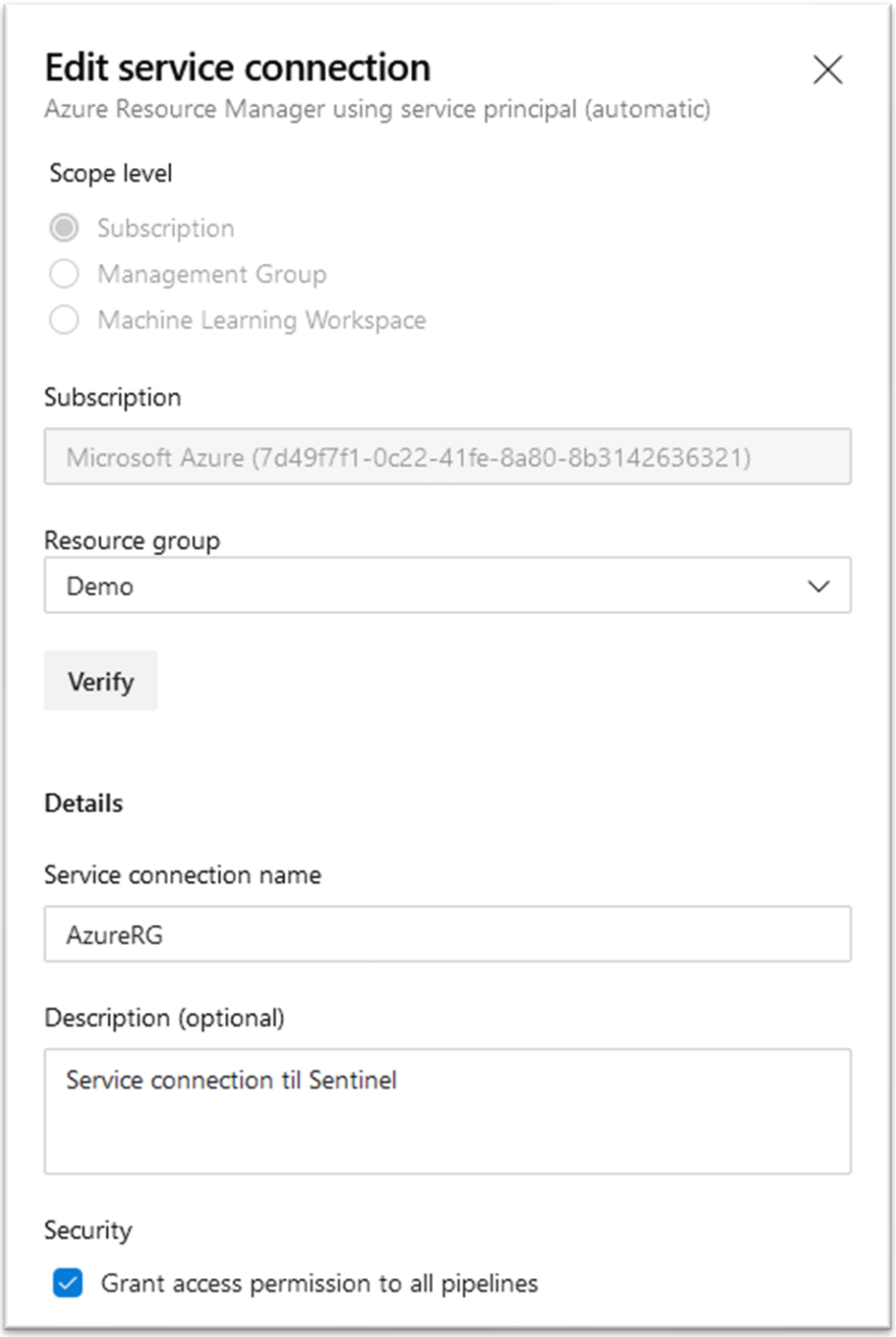
3.2.3 Opprette en Resource Manager

For å tillate DevOps til å kommunisere med Sentinel må vi sette opp en service connection av typen resource manager. For å gjøre dette går du til menyen project settings nederst på siden,



og går til fanen service connections.

Trykk på create service connection og velg så Azure Resource Manager i menyen. For enkelthets skyld velger vi service principal (automatic). Dette vil ta i bruk automatiske sikkerhetsinnstillinger med en service principal, som brukes til å få tilgang til ressurser i Azure. Dette vil bruke kredentialene fra brukerkontoen du er autentisert med i Azure. Dersom det er behov for å administrere tilganger via Azure AD, for eksempel om en har flere tenants, bør en velge manual for å opprette en managed service principal for å kunne delegere rettigheter.

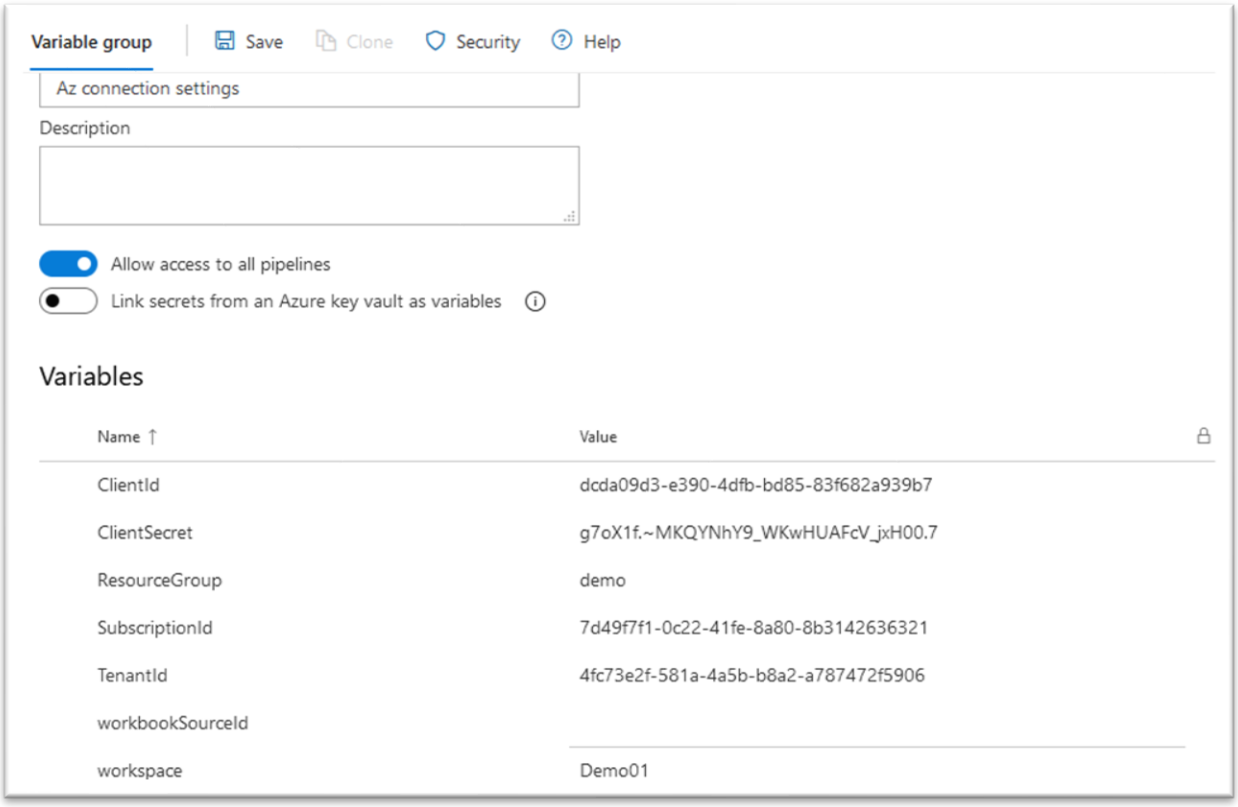


Viktig: Noter deg navnet du gir denne service connectionen, navnet vil bli brukt i konfigurasjonen av pipelines.

Opprette variabelgruppe

Scriptene som brukes for deployment til Sentinel tar i bruk flere variabler. I stedet for å manuelt oppdatere hvert enkelt script (spesielt relevant dersom en deployer til flere tenants), oppretter vi heller en variable group som scriptene kan referere til.

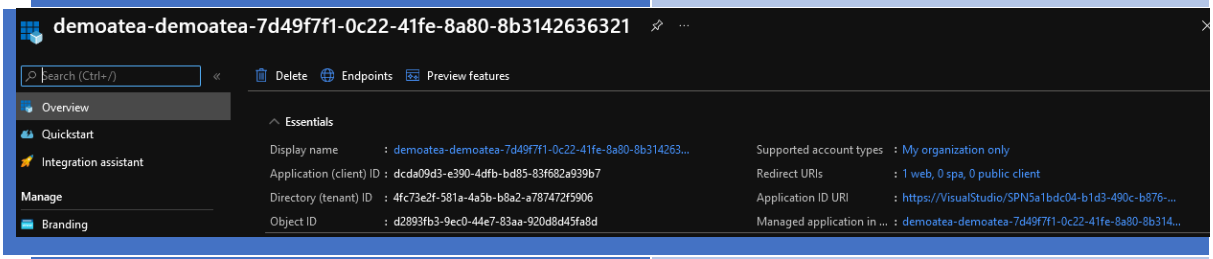
Variabelgrupper finner du under Pipelines -> Library, trykk her på + Variable group.



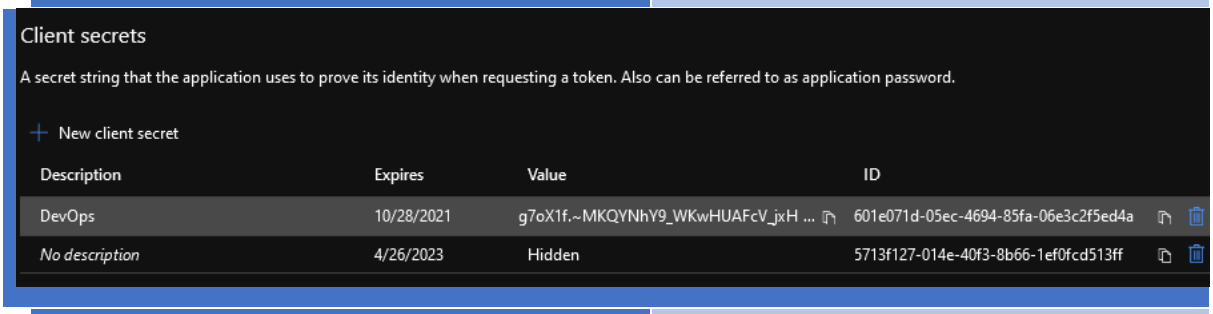
Vi skal gå igjennom variablene og vise hvor du finner hver variabel.

Viktig: Navnet du gir denne gruppen vil bli referert i pipeline-konfigurasjonen, så skriv det ned.

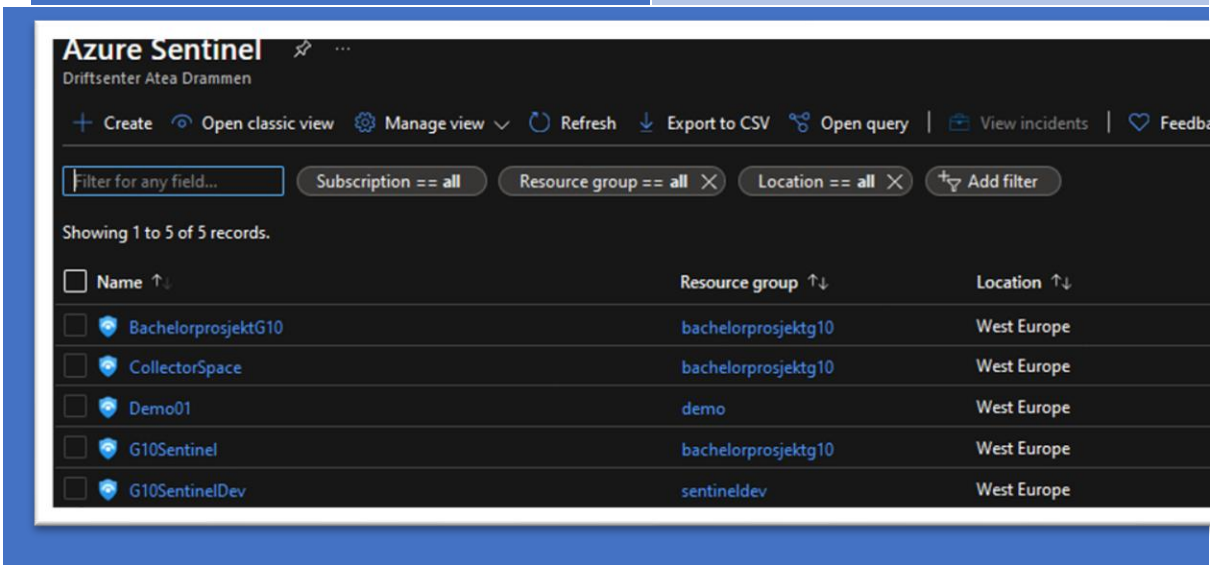
Navn	Forklaring
Client Id	Logg inn i Azure og søk opp app registrations. Her vil du finne resource manageren du opprettet i forrige punkt.



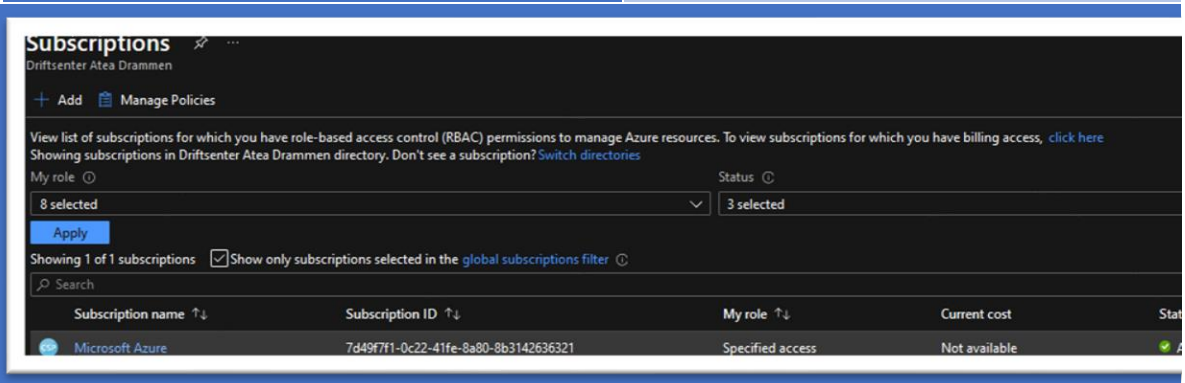
Client Secret	I app manager, gå til certificates og secrets. Trykk så på new client secret. Kopier og lim inn fra value-feltet.



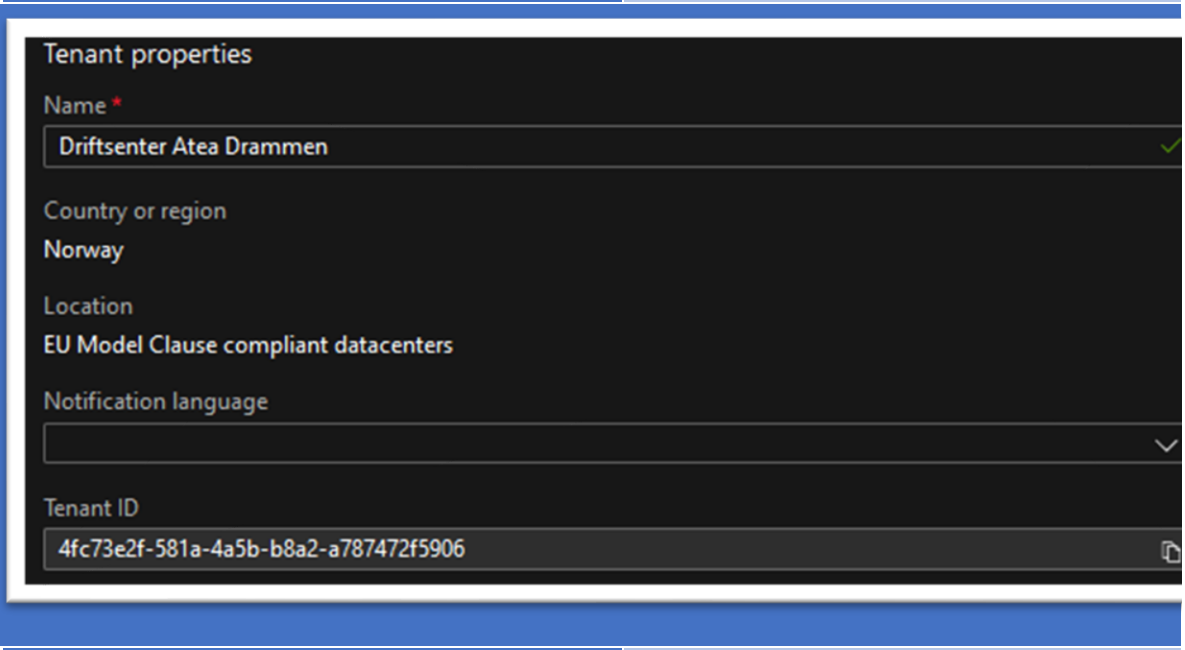
ResourceGroup	Ressursgruppa Sentinel ligger i. Dette finner du i Sentinel.



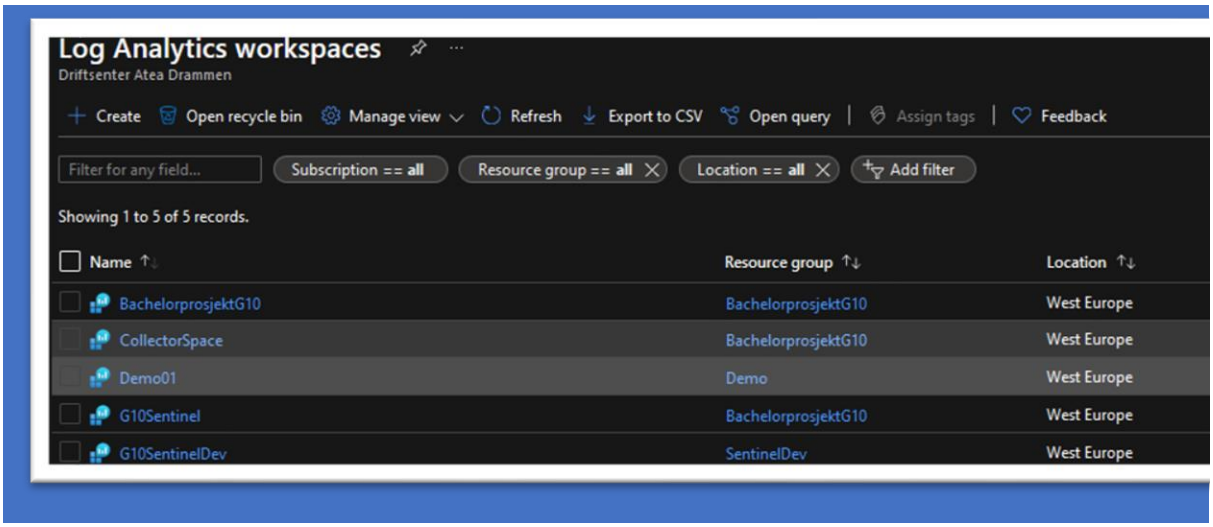
SubscriptionId	Denne finner du under subscriptions i Azure.



TenantId	Denne finner du i tenant properties i Azure.



workbookSourceId	Dette finner du i Sentinel etter å ha opprettet en workbook.
Workspace	Dette refererer til et Sentinel workspace, som du finner i Azure.

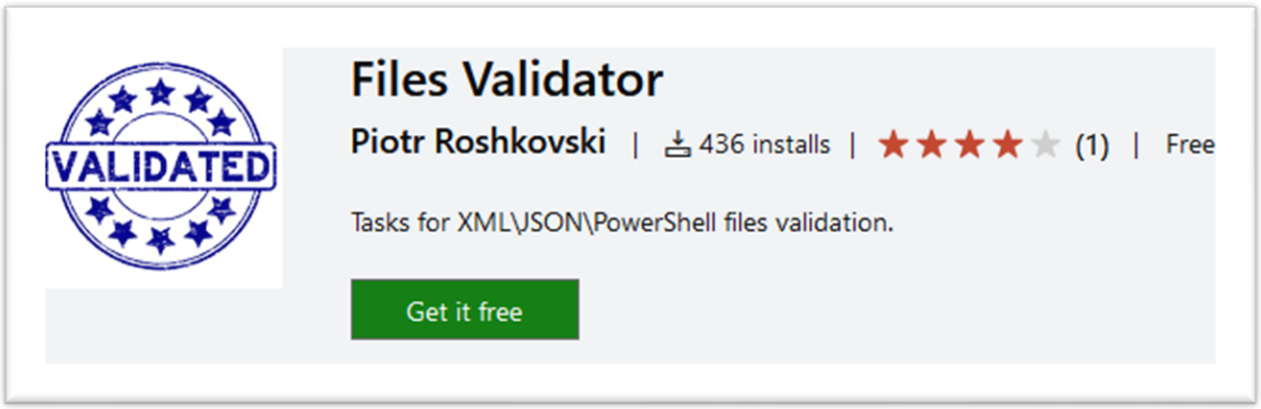


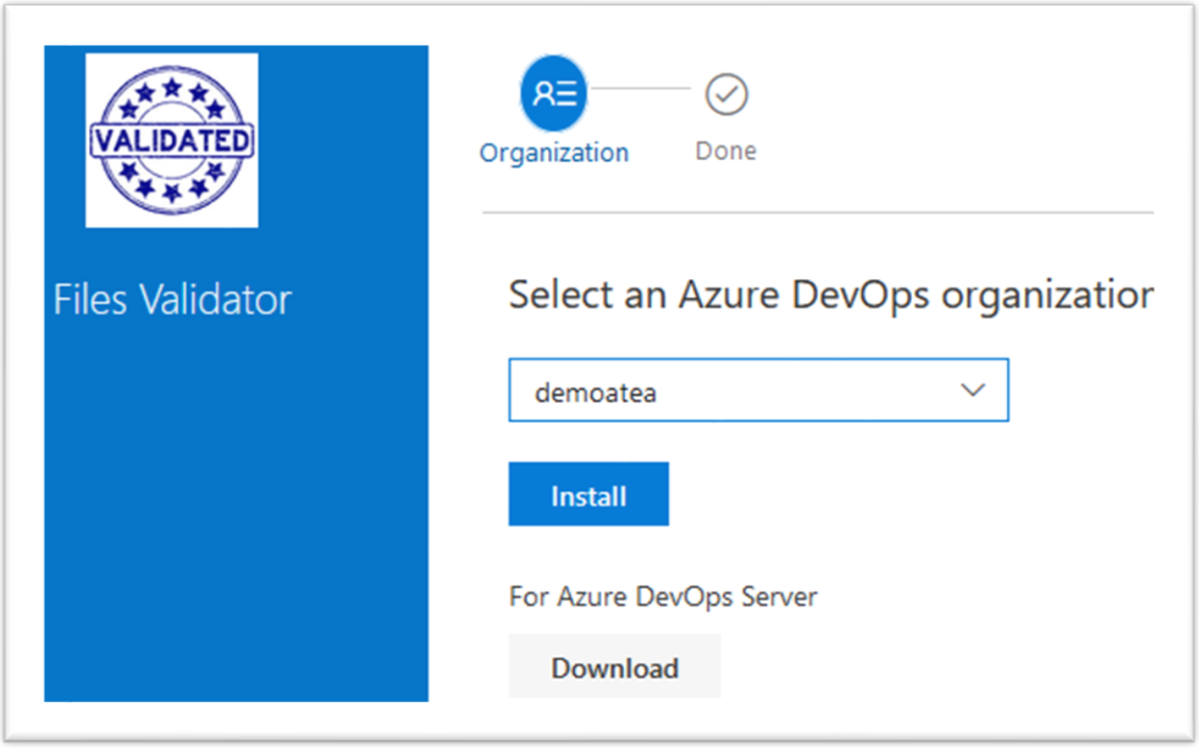
Klargjøring av pipelines

For å bygge og rulle ut endringer i koden til Sentinel, bruker vi pipelines. Pipelines er konfigurert med YAML-filer som vi kan lagre i repoen. Vi tar i bruk en pipeline for hver komponent i Sentinel som skal automatiseres. YAML-filene inneholder i tillegg til deployment av JSON-filene, et Powershell for å validere syntaxen av filene som skal deployes.

I tillegg har vi en pipeline for scripts, denne deployer ingenting til Azure, men lager en artifact som kan brukes av deployment scripts. Artifacts er filer som inneholder variabler og referanser til andre scripts som blir brukt under deployment.

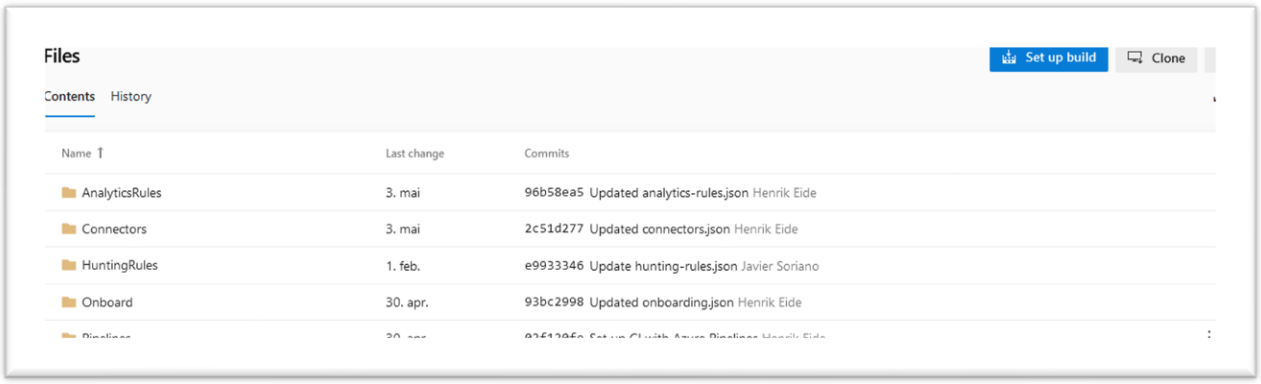
Viktig: Før du kjører din første pipeline, må du installere en add-on som vi bruker for å validere syntaxen på filene våre. Last den ned her i fra: https://marketplace.visualstudio.com/items?itemName=roshkovski.Files-Validator.





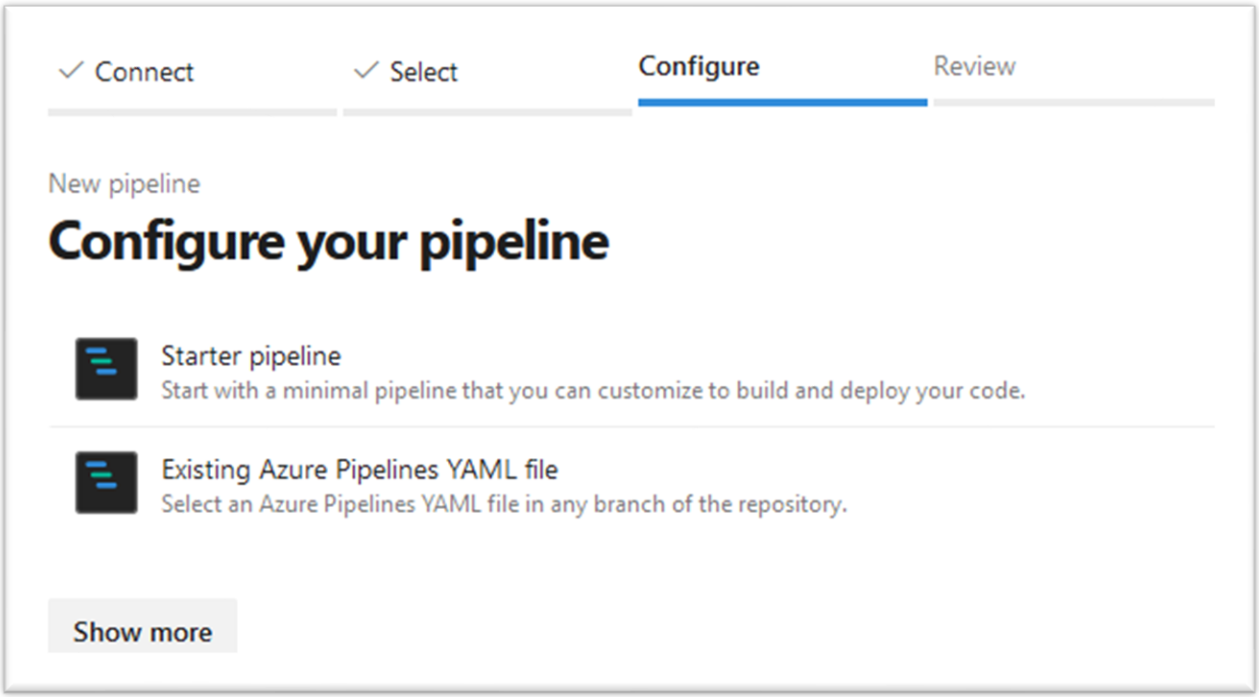
3.2.5 Oppsett av scripts pipeline

Viktig: Siden 1. mars 2021 er ikke parallel jobs inkludert i Azure DevOps prosjekter. For å søke om å få tilgang til parallel jobs gratis kan en søknad med navn og organisasjon sendes til azpipelines-freetier@microsoft.com.



Vi starter med å opprette en pipeline for scripts, denne vil bli brukt til å publisere en script-artifact som brukes av andre pipelines.

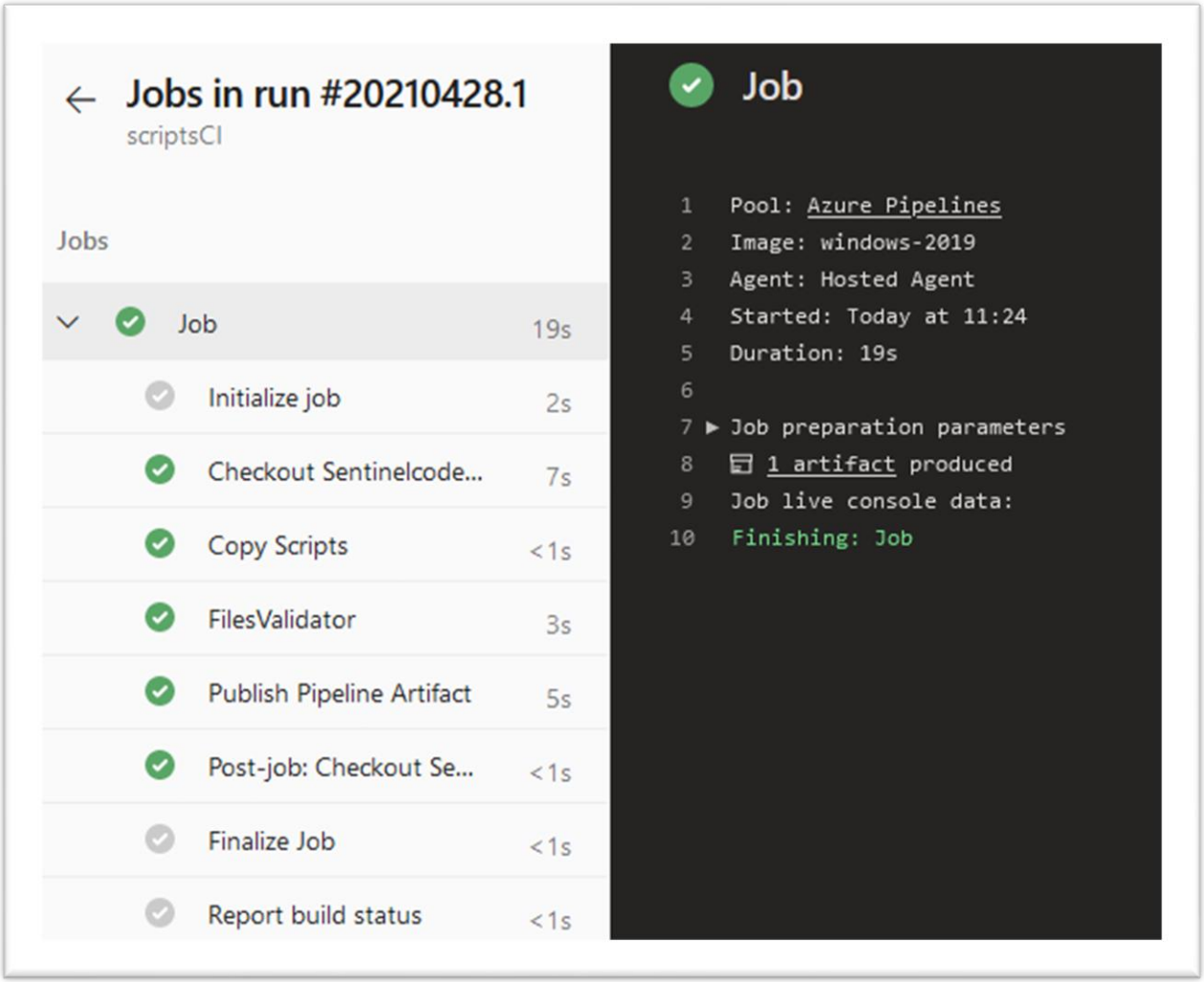
Gå til repoet og trykk på set up build. Du skal komme til følgende skjerm:



Her velger vi existing YAML-file og finner /Pipelines/buildScripts.yml.

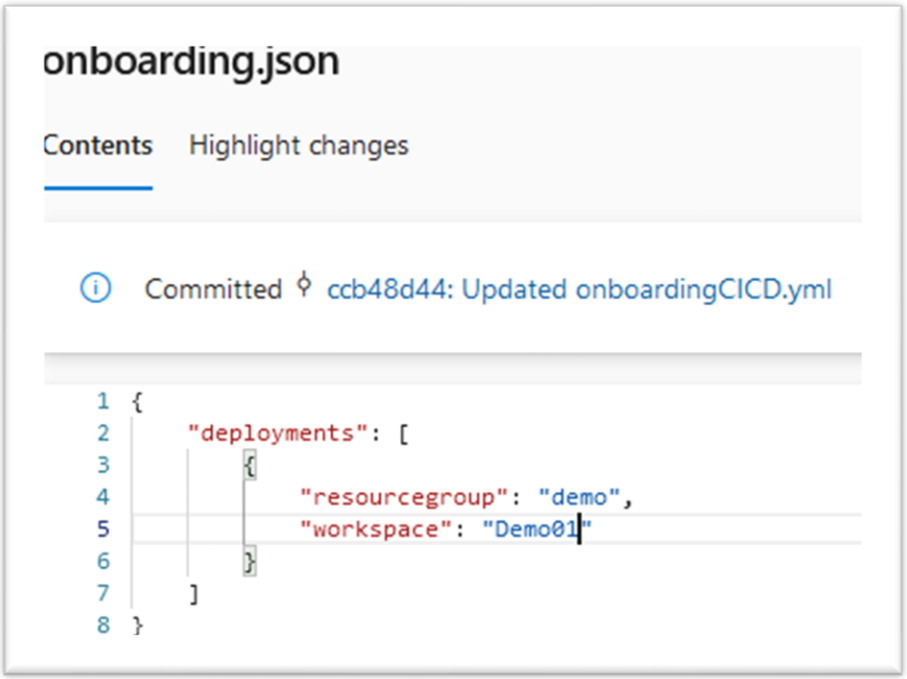
```yaml
# Copies script files to the agent and publishes an artifact with them

trigger: #Når det blir gjort endringer i mappen "Scripts", vil pipelinen bli aktivert.
 paths:
   include:
     - Scripts/*

pool:
  vmImage: 'windows-2019'

steps: #Kopierer innholdet i mappen "Scripts" til en artifact. Validerer syntaxen til filene og publiserer artifacten
- task: CopyFiles@2
  displayName: 'Copy Scripts'
  inputs:
    SourceFolder: Scripts
    TargetFolder: '$(build.artifactstagingdirectory)'
- task: Files-Validator@1
  inputs:
    rootDir: '$(build.artifactstagingdirectory)/*.ps1'
    validateXML: false
    validateJSON: false
    validateYAML: false
    validatePS: true
- task: PublishPipelineArtifact@1
  displayName: 'Publish Pipeline Artifact'
  inputs:
    targetPath: Scripts
```

Slik ser scripts-pipelinen ut. Denne pipelinen har tre trinn (tasks). Først kopierer den alle filene i scripts, og bruker utvidelsen vi installerte for å validere filene, deretter publiserer den en artifact som gjøres tilgjengelig for andre pipelines.


Jobs in run #20210428.1
scriptsCI
Jobs
Job 19s
Initialize job 2s
Checkout Sentinelcode... 7s
Copy Scripts <1s
FilesValidator 3s
Publish Pipeline Artifact 5s
Post-job: Checkout Se... <1s
Finalize Job <1s
Report build status <1s
Job
1 Pool: Azure Pipelines
2 Image: windows-2019
3 Agent: Hosted Agent
4 Started: Today at 11:24
5 Duration: 19s
6
7 ▶ Job preparation parameters
8 1 artifact produced
9 Job live console data:
10 Finishing: Job

Aktivering av Sentinel

Nå skal vi jobbe litt med onboarding av connectors og aktivering av Sentinel via DevOps.



Vi starter med å redigere onboarding.json. Denne finner du under mappen onboarding. Her er det lagt opp til at en kan installere/aktivere Sentinel i flere workspaces på samme tenant. Dette åpner for å ha flere environments, slik som å deploye til ulike development- og production workspaces. Dersom du ikke ønsker dette enda, fyller du enkelt inn navnene på ressursgruppen og workspacet du opprettet tidligere.

Deretter klargjør vi pipelinen ved å redigere onboardingCICD.yml, denne finner du under pipelines-mappen. Denne pipelinen aktiverer sentinel i de workspaces som er angitt av onboarding.json. Vi redigerer linje 59 (azureSubscription) til det samme navnet som vi ga til vår Resource Manager tidligere. På linje 50 redigerer vi group til å være det samme navnet som vi tidligere ga vår variabelgruppe. Deretter kan du gå til pipelines og starte utrullingen. Denne pipelinen vil aktivere Sentinel, dersom det ikke allerede er gjort.

Aktivering av connectors

Dette er siste steget før vi kan sette i gang med å rulle ut kode til Sentinel. For å automatisere aktiveringen av connectors må vi ta i bruk Sentinel APIen. Scriptet Import-AzSentinelDataConnector.ps1 i scripts-mappen henter listen over connectors vi vil aktivere fra connectors.json og bruker EnableConnectorsAPI.ps1 fra AzSentinel for å aktivere connectors.

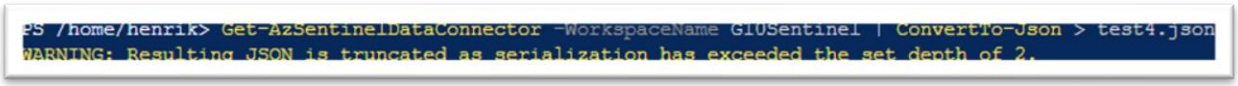
Det er kun mulig å automatisere «Microsoft-native» connectors, eller et fåtall andre som er forhåndsdefinerte i Sentinel. Dette er fordi eksterne connectors behøver enten en agent, eller bruker CEF (Common Event Format), som betyr at en må knytte den opp mot en log collector. En liste over hvilke connectors som kan aktiveres via API finner du her: https://docs.microsoft.com/en-us/azure/sentinel/connect-data-sources.

For å legge til flere connectors kan en enkelt legge til flere elementer i connectors.json. Den enkleste måten å gjøre dette på er å først manuelt aktivere connectors i Sentinel, og så bruke Azure CLI for å hente ut konfigurasjonen gjennom APIen.

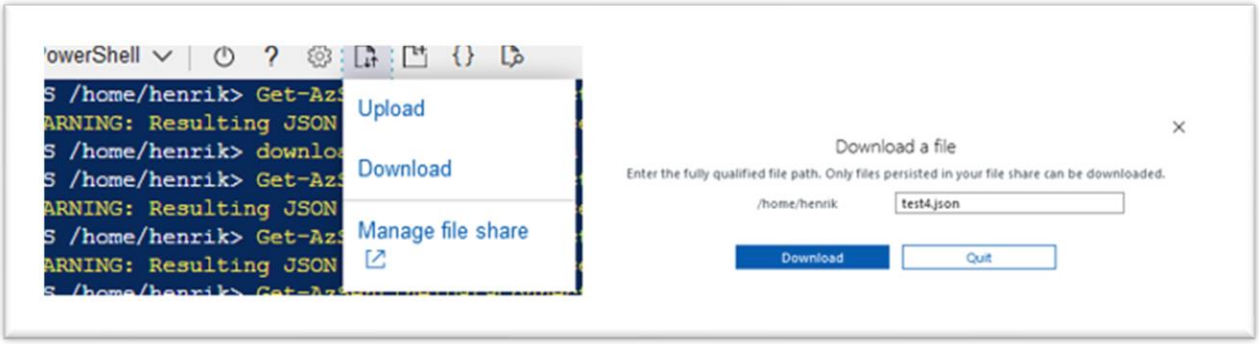
Eksempel på henting av connectors fra et Sentinel workspace:

I dette workspacet har vi manuelt aktivert to connectors. Målet med denne demonstrasjonen er å hente ut disse to i JSON-format, for å så legge de i DevOps slik at de kan deployes til andre Sentinel workspaces.

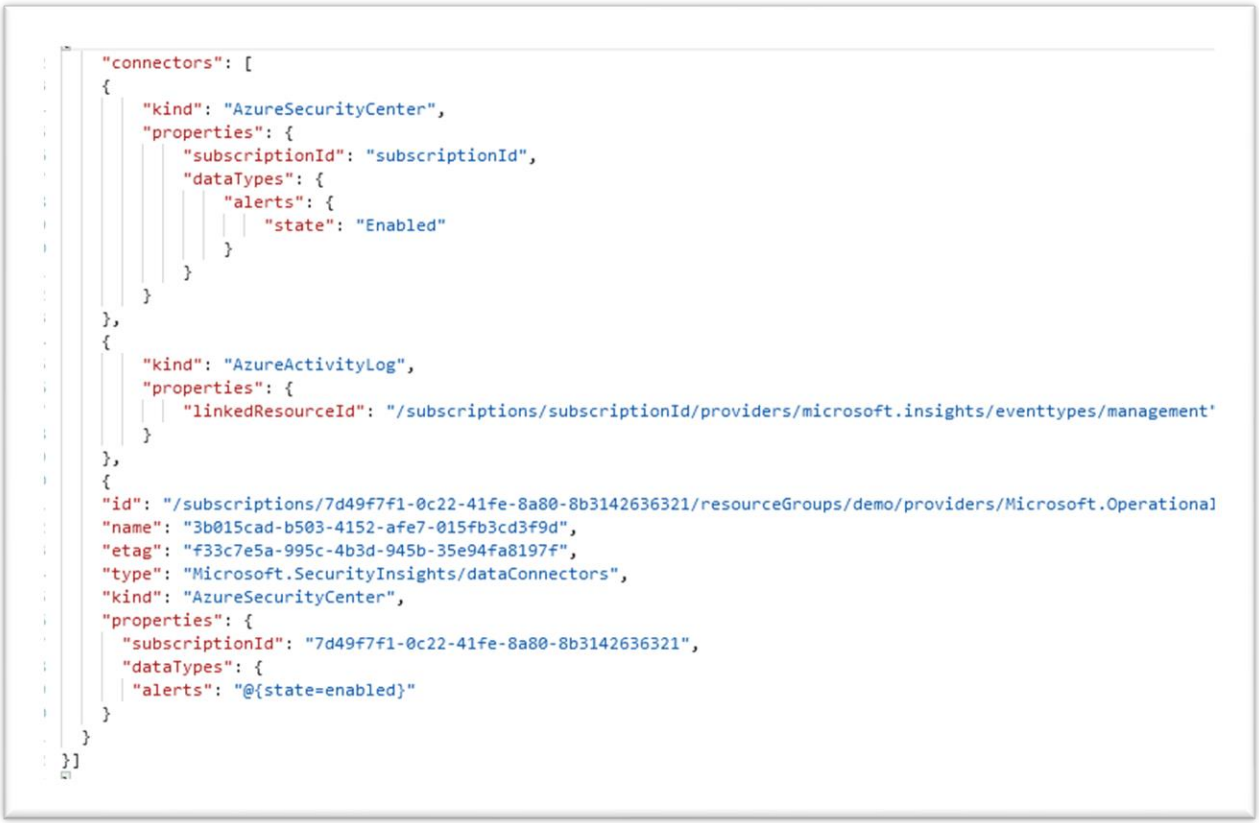
Vi starter med kommandoen Get-AzSentinelDataConnector. Her kreves det en variabel, «WorkspaceName». Deretter sender vi utdataen til ConvertTo-Json metoden for å få det på riktig format. Gi filen et fornuftig navn. Advarselen kan ignoreres.



For å laste ned filen trykker du på knappen på verktøyslinjen og så download. Skriv inn filnavnet.



Den resulterende tekstfilen kan så enkelt limes direkte inn i connectors.json i DevOps. Men ta vare på AzureSecurityCenter og AzureActivityLog, for dette refererer til selve Sentinel.



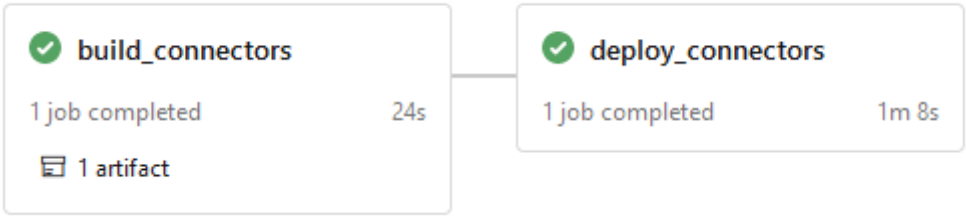
```
"connectors": [
{
    "kind": "AzureSecurityCenter",
    "properties": {
        "subscriptionId": "subscriptionId",
        "dataTypes": {
            "alerts": {
                "state": "Enabled"
            }
        }
    }
},
{
    "kind": "AzureActivityLog",
    "properties": {
        "linkedResourceId": "/subscriptions/subscriptionId/providers/microsoft.insights/eventtypes/management'
    }
},
{
"id": "/subscriptions/7d49f7f1-0c22-41fe-8a80-8b3142636321/resourceGroups/demo/providers/Microsoft.Operational
"name": "3b015cad-b503-4152-afe7-015fb3cd3f9d",
"etag": "f33c7e5a-995c-4b3d-945b-35e94fa8197f",
"type": "Microsoft.SecurityInsights/dataConnectors",
"kind": "AzureSecurityCenter",
"properties": {
  "subscriptionId": "7d49f7f1-0c22-41fe-8a80-8b3142636321",
  "dataTypes": {
   "alerts": "@{state=enabled}"
  }
 }
}]
```

Oppsett av connectors pipeline

```
41  - stage: deploy_connectors
42    jobs:
43      - job: AgentJob
44        pool:
45          name: Azure Pipelines
46          vmImage: 'windows-2019'
47        variables:
48        - group: Az connection settings
49        steps:
50        - download: current
51          artifact: ConnectorsFile
52        - download: Scripts
53          patterns: '*.ps1'
54        - task: AzurePowerShell@4
55          displayName: 'Create and Update Connectors'
56          inputs:
57            azureSubscription: 'AzureRG'
58            ScriptPath: '$(Pipeline.Workspace)/Scripts
59            ScriptArguments: '-TenantId $(TenantId) -C
60            azurePowerShellVersion: LatestVersion
61            pwsh: true
```

Vi åpner connectorsCICD.yml og redigerer group (linje 48) til å korrespondere med variabelgruppen. Vi redigerer også azureSubscription (linje 57) til å korrespondere med Resource Manageren vi opprettet.

Deretter kan du kjøre connectorsCICD-pipelinen og sjekke i Azure at alt fungerte.



4. Utførelse

Til nå har vi aktivert opprettet workspaces og aktivert Sentinel, aktivert connectors og satt opp grunnleggende pipelines. I dette kapittelet går vi videre med mer praktisk bruk av Sentinel. Vi går gjennom oppsett av en log collector, som brukes til å samle inn logger fra systemer som ikke støttes av Sentinel. Vi går også inn på hvordan en skriver og ruller ut ulike typer regler for Sentinel. Dette gjelder i hovedsak playbooks, analytics- og hunting rules.

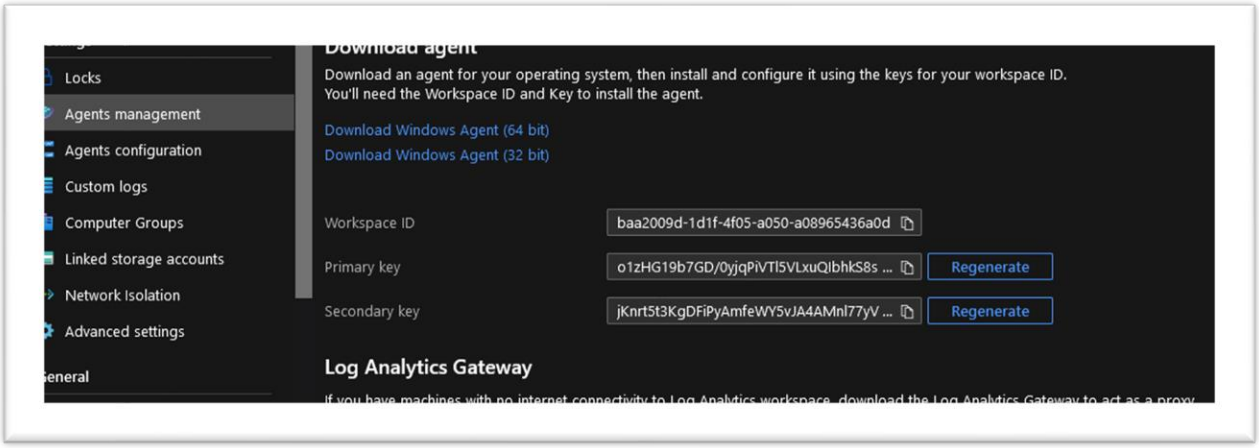
Deretter ser vi på hvordan en systemet reagerer på at en regel blir brutt, og hvordan vi ved bruk av Azure Automation kan skrive playbooks som automatisk responderer på regler som får treff.

4.1 Oppsett av Log Collector til Azure Sentinel

Vi skal nå vise hvordan vi kan sette opp en log collector for å samle data fra kilder som ikke har en native connector direkte til Azure Sentinel. En slik log collector sender og mottar loggdata og standardiserer formatet på disse før det sendes videre til Sentinel for analyse og kontroll.

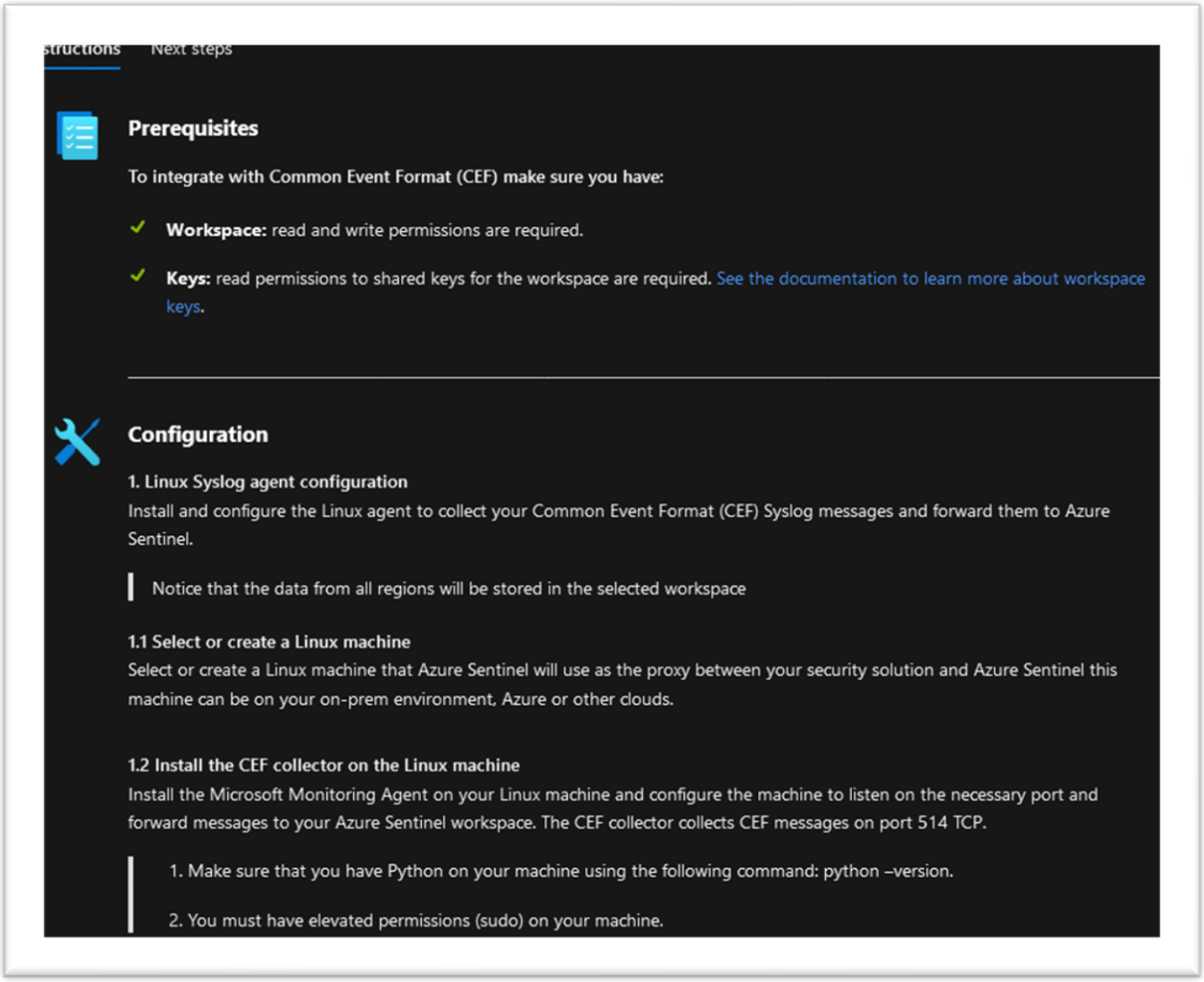
For å sette opp en log collector er det første man trenger en linux server med sudo-rettigheter. Når dette er gjort må man sjekke at man har Python versjon 2.7 eller 3 installert. Det er også viktig at linux serveren ikke er koblet til noe annet Azure workspace før man kjører scriptet under. Det er også viktig at serveren har minimum 4CPU kjerner og 8GB Ram for å kunne utføre operasjonene som kreves på en god og effektiv måte. Med disse spesifikasjonene vil en log collector bestående av en maskin kunne håndtere 8500 innhentede events per sekund. (EPS)

For videre konfigurering er det viktig å ha Workspace Iden og primærnøkkelen (primary key) tilgjengelig disse finner du her **Azure Sentinel -> Trykk på workspace navnet -> Settings -> Workspace settings** også trykker vi på **Agents management** Da vil du komme til en side som ser slik ut.

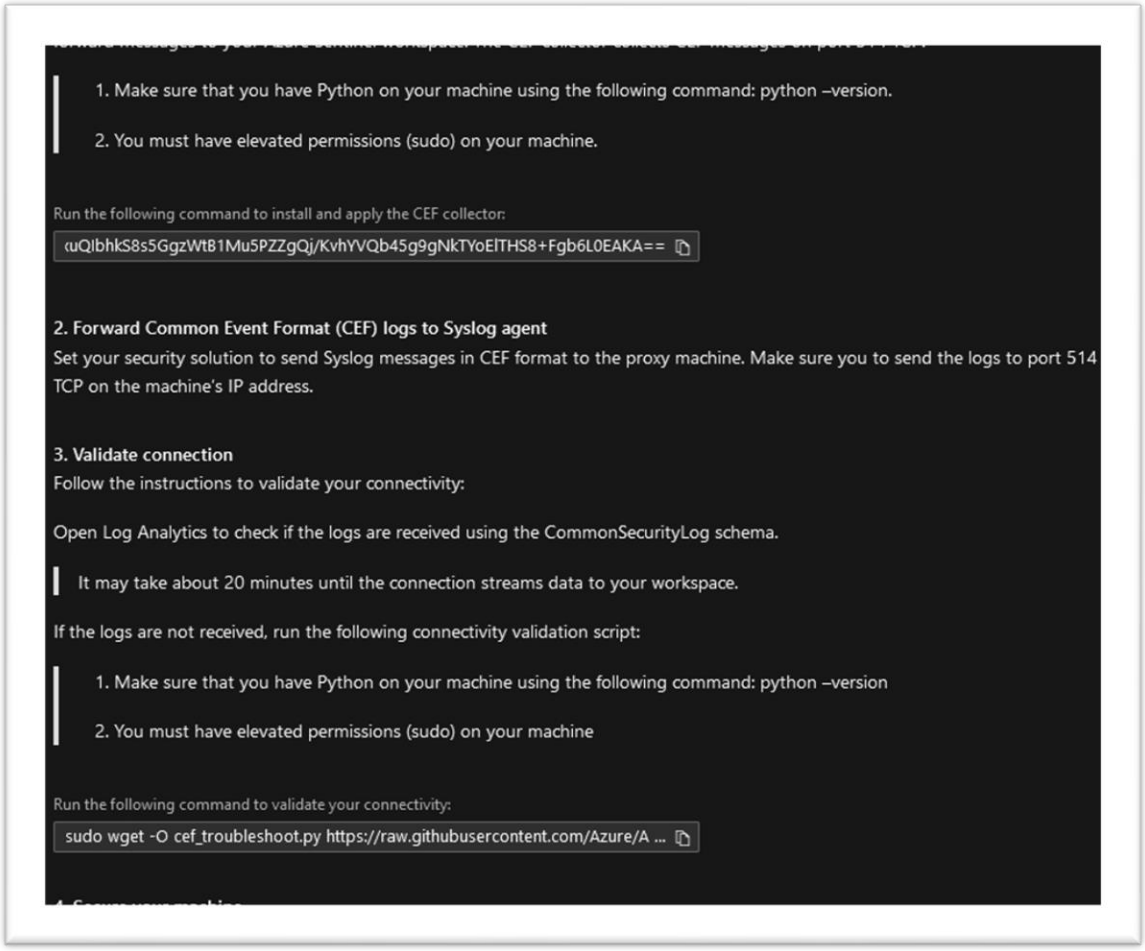


Her finnes både Workspace ID og Primærnøkkelen på en og samme side og dette gjør konfigureringen vår enklere når vi skal kjøre scriptet på neste side.

I Azure Sentinel siden inne på azure ligger workspacet der Sentinel brukes. Her går vi videre inn til **data connectors** skriver CEF (Common Event Format) og trykker på denne. Nede i høyre hjørnet vil du da se en knapp som sier **Open Connector page**.

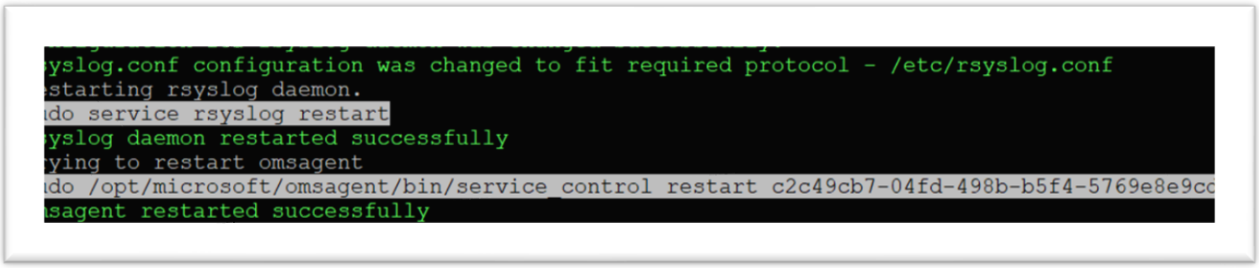


Her er det viktig at du har riktige rettigheter, som vi ser av skjermbildet ovenfor stemmer rettighetene våre med det vi trenger for å gå videre.




```
sudo wget -O cef_installer.py https://raw.githubusercontent.com/Azure/Azure-Sentinel/master/DataConnectors/CEF/cef_installer.py&&sudo python cef_installer.py baa2009d-1d1f-4f05-a050-a08965436a0d
```

Her er kommandoen som blir brukt i sin helhet for å installere og konfigurere CEF collectoren. Når denne kjører uten feil ser det slik ut.



Det dette scriptet gjør er å installere Log Analytics agenten (OMS agent) og konfigurerer den til å lytte etter CEF meldinger fra Linux Syslog Daemon på port 25226. For så å videresende de mottatte loggene sikkert over TLS til Azure Sentinel Workspacet.

Siden konfigureres Linux Syslog Deamon til å lytte etter syslog pakker fra sikkerhetsløsninger over TCP på port 514. Så videresender den meldingene den identifiserer som CEF til log analytics agenten på local host ved å bruke TCP port 25226.

Når du kjører scriptet kan det være du får en feilmelding om mappingen av computer feltet. Da blir du bedt om å kjøre denne komandolinjen for å rette på dette.
grep -i "'Host' => record\['host'\]" /opt/microsoft/omsagent/plugin/filter_syslog_security.rb
dette sjekker integriteten på hvor mappingene blir lagret, dersom det er en feil kjører vi denne.

```
sed -i -e "/'Severity' => tags\[tags.size - 1\]/ a \ \t 'Host' => record['host']" -e "s/'Severity' => tags\[tags.size - 1\]/&,/" /opt/microsoft/omsagent/plugin/filter_syslog_security.rb && sudo /opt/microsoft/omsagent/bin/service_control restart [workspaceID]
```

Når dette er gjort kan vi stille inn log collectoren slik at den sender tidsdata om når hendelsen faktisk oppstod og ikke når den ble sendt fra log collectoren til sentinel. Dette gjøres ved å kjøre to enkle kommandoer.

For å deaktivere synkroniseringen med sentinel agenten:

```
sudo su omsagent -c 'python
```

For å sette tidspunktet vi ønsker {ws_id} må erstattes med den aktuelle Iden:

```
Sudo wget -O TimeGenerated.py https://raw.githubusercontent.com/Azure/Azure-Sentinel/master/DataConnectors/CEF/TimeGenerated.py && python TimeGenerated.py
```

For å sjekke at alt fungerer kjører vi et validiation script:

```
sudo wget -O cef_troubleshoot.py https://raw.githubusercontent.com/Azure/Azure-Sentinel/master/DataConnectors/CEF/cef_troubleshoot.py&&sudo python cef_troubleshoot.py
```

og ser om det kjører uten feilmeldinger. Igjen erstatter vi workspace iden med vår id og ser hvordan det ser ut. Det kan ta opptil 20 minutter fra opprettelse av en log collector til den begynner å sende data til Sentinel. Av og til kan tålmodighet være den beste løsningen!

En slik type log collector kan også være hensiktsmessig å bruke dersom organisasjonen er i endring fra en on-premise løsning til skyløsning, man kan ta i bruk Sentinel selv om systemene er fysiske ved å sende dataen fra disse via en eller flere log collectors. Dette vil være spesielt aktuelt for Active Directory, Microsoft Defender og Endpoint Protection.

Dette viser også en del av den sammensatte årsaken til hvorfor man skal velge Sentinel som SIEM-løsning. Den er skalerbar, man kan koble til nesten alt og man administrerer ikke egen maskinvare, noe som sparer store kostnader.

4.3 Aktivere connectors

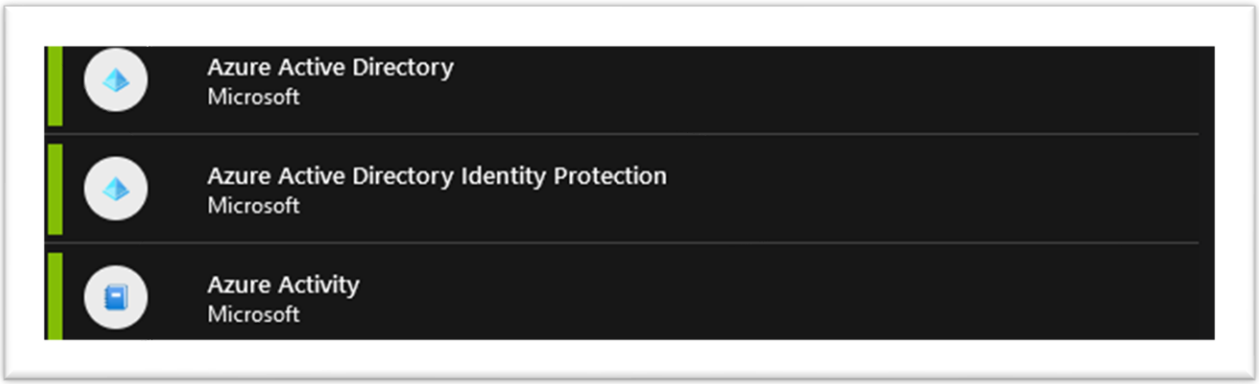
4.3.1 Microsoft native connectors

Azure har mange connectors i sin portefølje. Mange av disse tilhører også deres egne produkter. Disse connectorene finner man under **Sentinel** og -> **Data connectors** eksempler på disse er Microsoft 365 Defender, Azure active directory identity protection, og Azure Key vault. Disse connectorene kommer med detaljerte instruksjoner og gjør det enkelt å sette opp og konfigurere.

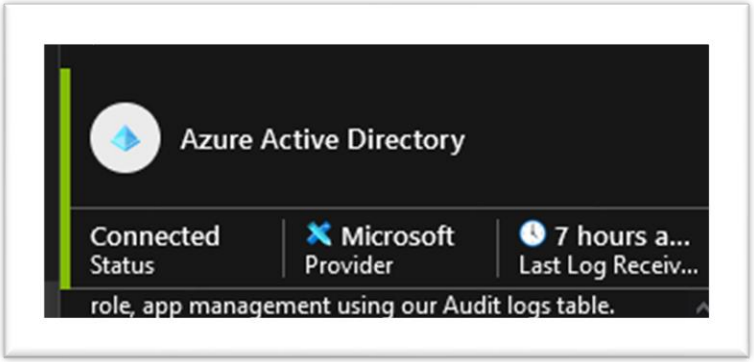
Det disse connectorene gjør er egentlig å forenkle oppsett og opprettelse av en tilkobling mellom miljøet i Azure og Sentinel til forskjellige datakilder. Fordelen med disse er at de er garantert kompatible, siden de er validert og testet. Det minimerer også sjansen for menneskelige feil og feilkonfigurasjoner da man i hovedsak ikke trenger å endre noe selv for at det skal fungere.

Det er også slik at Microsoft har et team med sikkerhetseksperter og utviklere som stadig forbedrer og tester forskjellig funksjonalitet for å kunne tilby det beste produktet til kundene. Bare siden vi begynte med bachelorprosjektet vårt har det kommet flere nye connectors og ny funksjonalitet som har hatt innvirkning på løsningen av oppgaven vår.

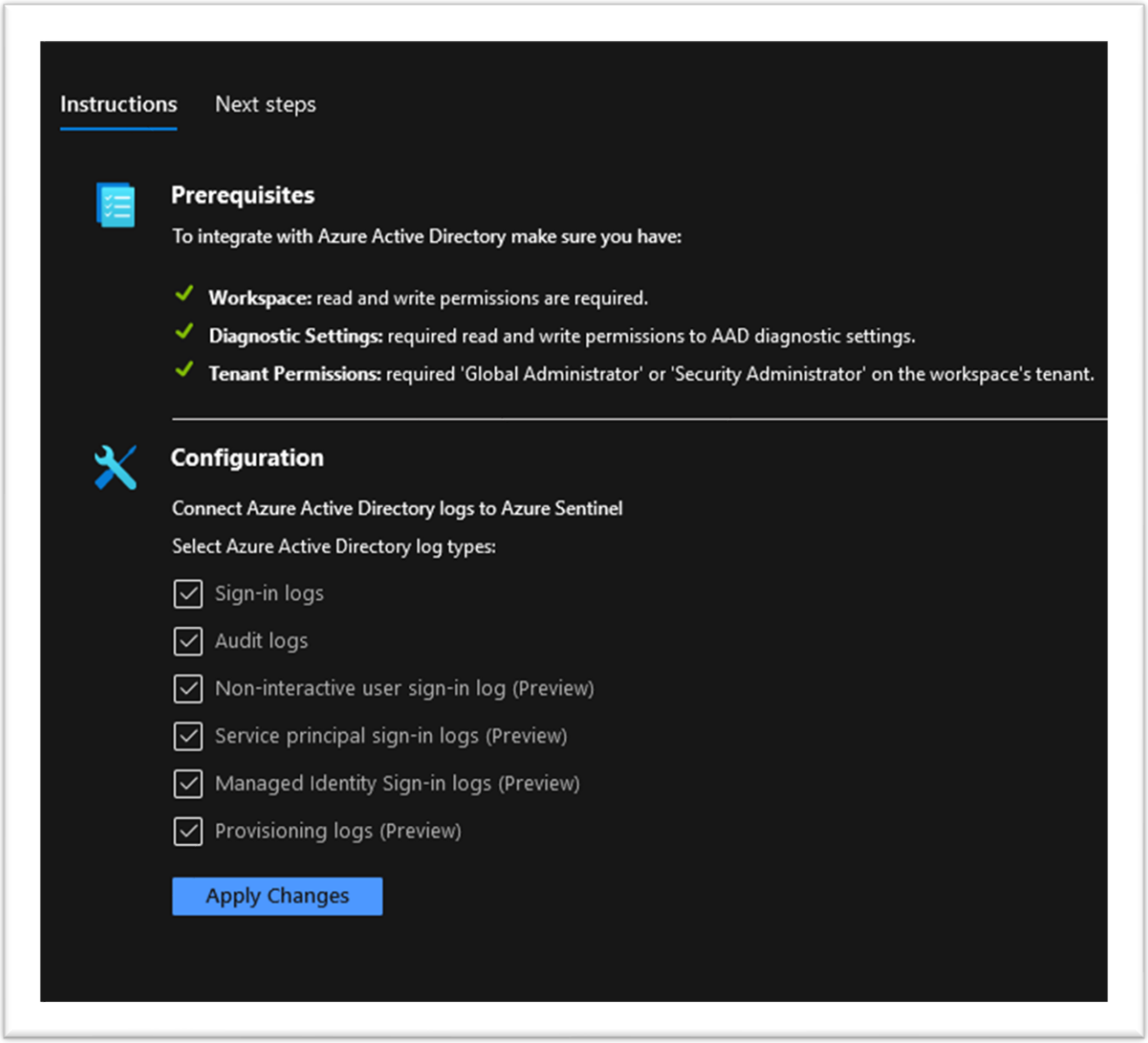
Det er dog noen connectors som er viktigere enn andre og de vi har valgt å fokusere på er basert rundt microsoft 365 og Azure, og begrenser oss til dette. For øyeblikket er det 103 mulige data connectors i Azure, men vi bruker bare noen av disse.



4. *4.3.2 Azure Active Directory*

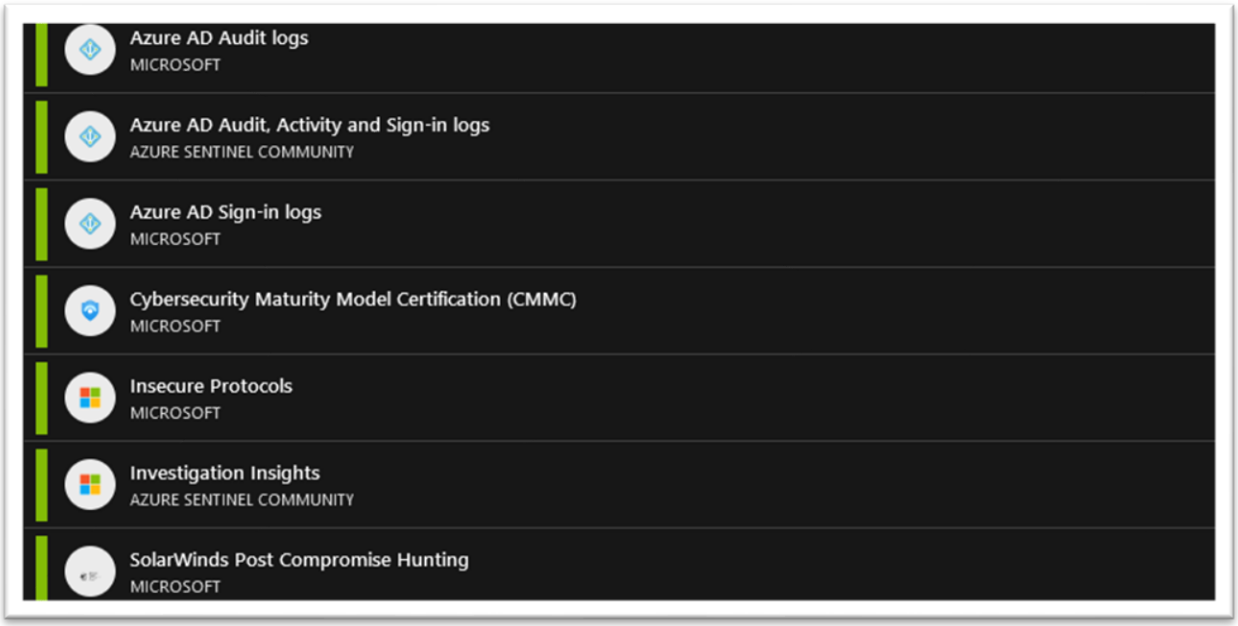


Vi ser at den er tilkoblet, at den er offisielt utgitt av microsoft og når siste data er sendt fra den til sentinel.



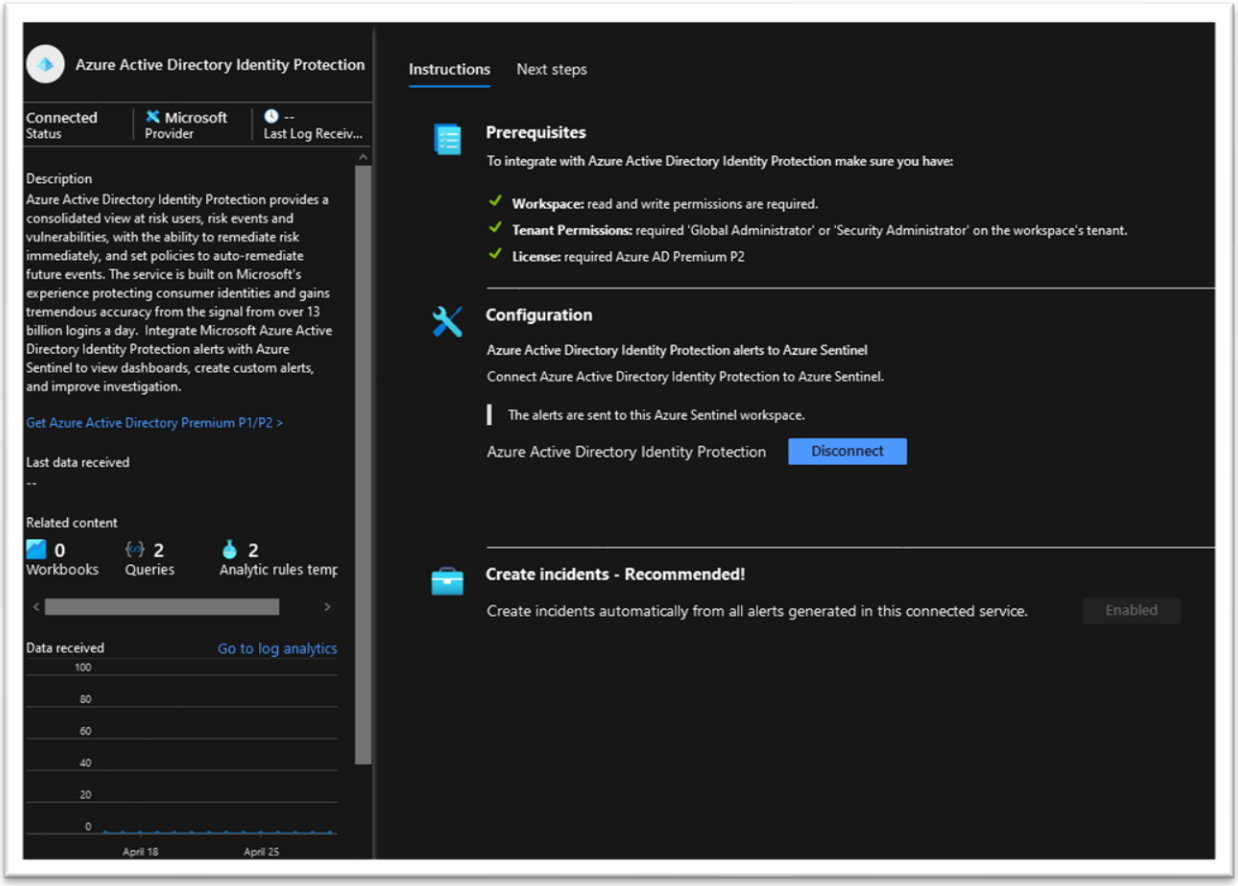
Her ser vi at Sentinel sjekker at man har tilstrekkelige rettigheter til å aktivere connectoren. Hva som trengs og hvilken konfigurasjon man ønsker for den spesifikke connectoren. Vi ser også at Microsoft gjør det så enkelt som mulig å konfigurere og endre uten noen form for kode og minimerer menneskelige feil.

Merk også at Sign-in logs krever en P1 eller P2 Azure AD subscription, ellers kan man bruke alle disse gratis så lenge datastrømmen ikke overstiger 8GB om dagen. Vi ser også at det er flere anbefalte playbooks å kjøre med denne connectoren.



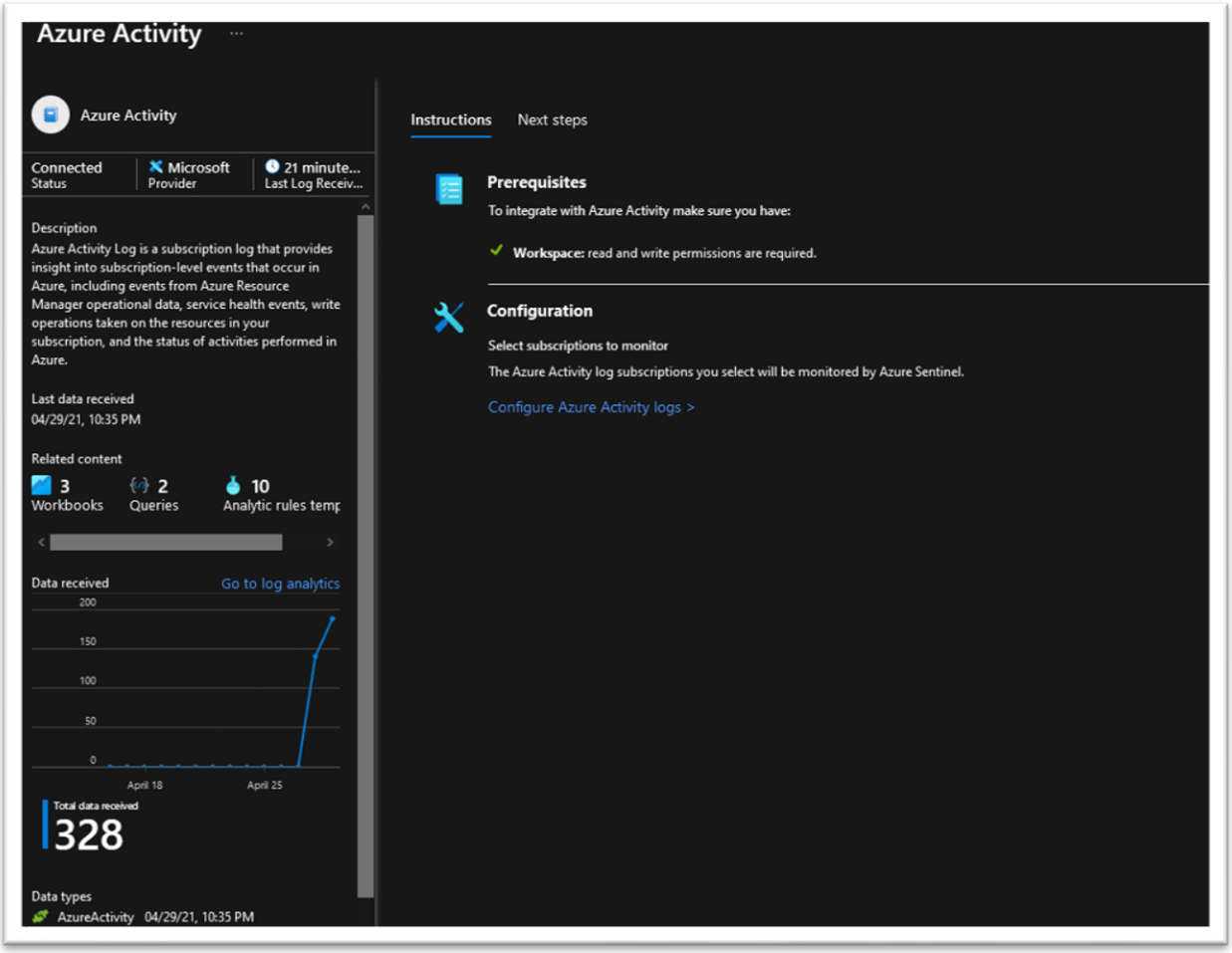
Lagrer og kjører disse workbookene sammen med connectoren. Vi ser også at det er 38 relevante alert rules som man kan kjøre med denne connectoren disse er gode grunnsteiner å bygge regler på slik at man slipper å skrive reglene helt fra bunnen av og gir også Microsoft en god mulighet til å rette oppmerksomhet mot sikkerhetsfeil så fort de oppdages ved å lage oppdaterte regler og templates. Et eksempel på dette er regler mot sårbarheten i Microsoft Exchange.

Azure Active Directory Identity Protection



Denne er veldig grei, ser at den er connected, kan deaktivere den like enkelt som vi aktiverte den og lar den generere hendelser etter hvert som de forekommer slik som er anbefalt av Microsoft, dette er to enkle klikk, men er helt nødvendig for at Sentinel skal motta data fra Azure AD.

4.3.3 Azure Activity



Sender log data om hva som skjer på subscription level i Azure. Dette er hendelser fra Ressursmanageren operasjonell data, hvor godt de forskjellige servicene leverer samt status på operasjoner som er gjort i Azure.

4.4 – KQL

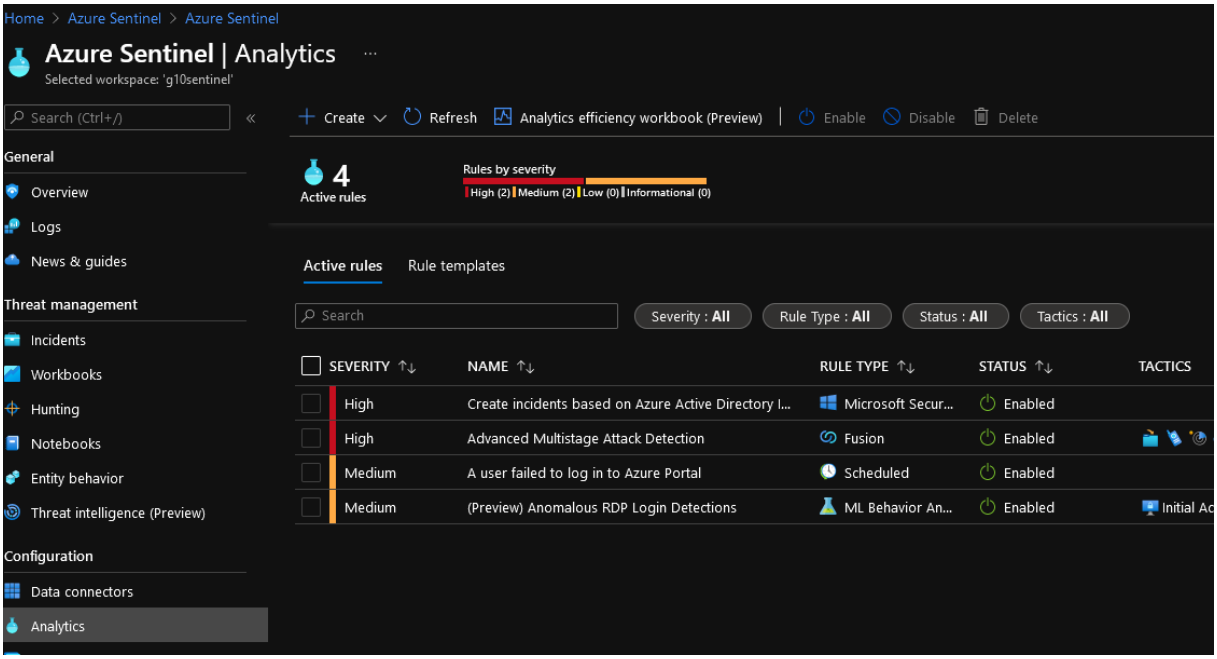
KQL eller populært kalt kusto er et read only skriptspråk brukt for å hente og prosessere data. Svarene man får er i plain text. Språket ble utviklet etter et enkelt prinsipp om at det skulle være enkelt å lese og enkelt å bruke. Det skal gi god ytelse og være skalerbart.

KQL brukes altså til å resultere tubulær data og brukes i Azure application insight, Azure log analytics, Windows defender advanced threat protection, Azure Security Center. Dette språket blir derfor meget viktig når man ønsker å analysere data fra Azure Sentinel med hendelser og trusler. KQL bruker også elementer av andre script språk og forstår allerede etablerte scriptspåk som SQL Python og andre.

4.5 Analytic rules

4.5.1 Analytic rules i Sentinel

Analytic rules er en funksjon man finner inne i Azure Sentinel. Etter å ha trykket på det aktuelle workspacet for oss G10Sentinel går man inn på **Analytics** i menyen til venstre under **Configuration**.



Da kommer man til en side som ser slik ut. Dette bildet viser 4 analytic rules som også demonstrerer de fire forskjellige regeltypene som finnes innenfor alert rules.

4.5.2 Forskjellige typer analytic Rules

Det finnes fire forskjellige typer analytic rules. Disse representerer altså alle de forskjellige typene regler og triggere man kan aktivere og forholde seg til innad i Sentinel. De er bedre kjent som “Scheduled analytic rules”, “Fusion rules”, “Security incidents” og “Machine learning and behaviour analytics”.

4.5.2.1 Scheduled analytic rules

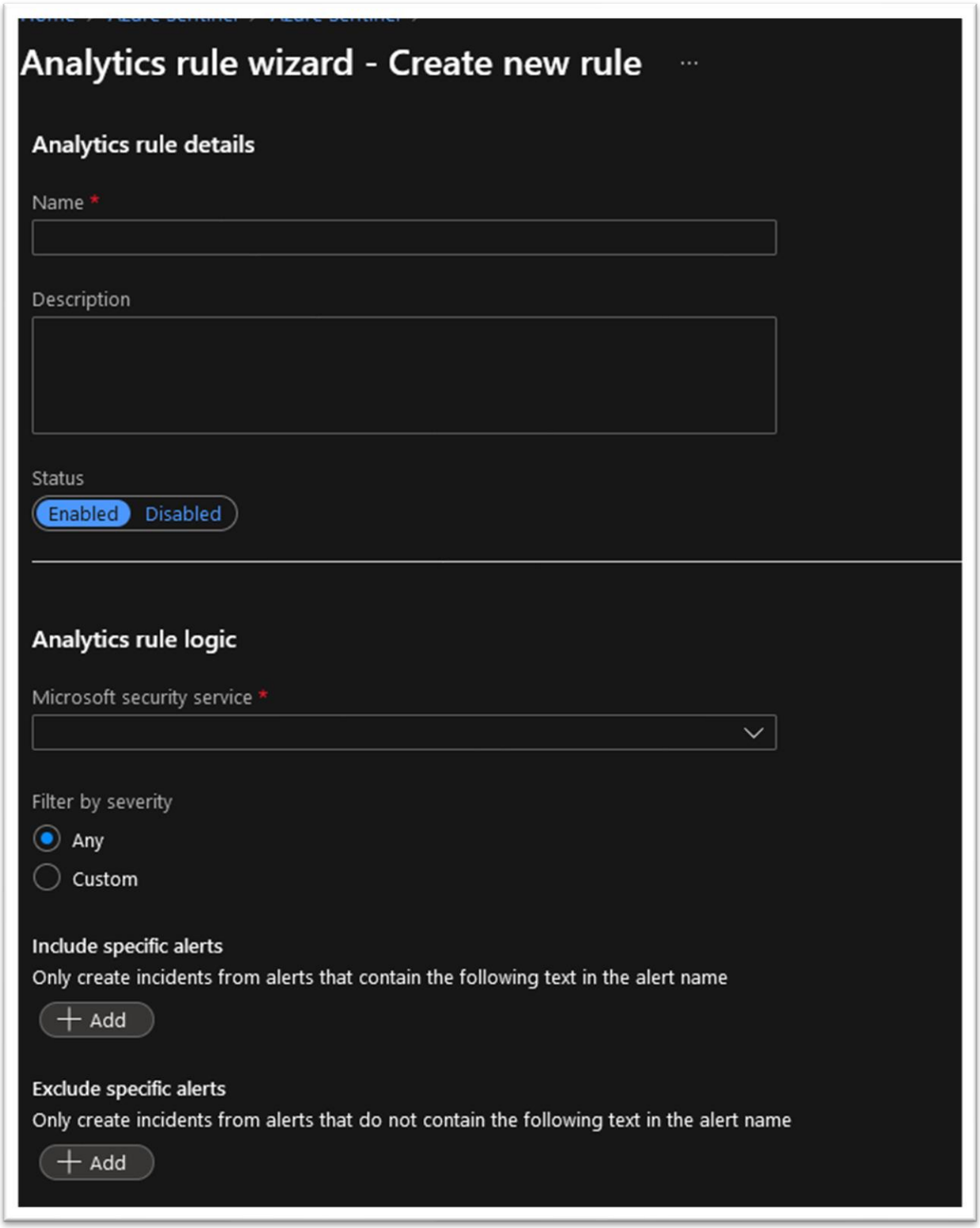
Scheduled analytic rules i Azure Sentinel er regler som kjøres i et visst mønster. Den regelen som er på bildet, er en scheduled regel som kjører hver 3. time og analyserer data fra de siste 3 timene. Der den altså snapper opp endringer som har skjedd innenfor et visst tidsrom. Den er ikke kontinuerlig, men kan stilles til alt fra å kjøre hvert 5 minutt til hver 5 måned.

4.5.2.2 Fusion rules

Dette er regler som bruker maskinlæring og fusion detection til å oppdage trusler som normalt flyr under radaren ved å analysere brukeres adferd og kartlegge mulige suspekte handlinger. Det at Azure sentinel har tilgang til så mye data og kan se mønstre gjør at man på grunnlag av unormale aktivitetsmønstre kan oppdage en trussel og dette gjøres med Fusion rules. Denne funksjonen er aktivert som standard fordi logikken bak er skjult og derfor ikke kan endres eller tilpasses, man kan bare lage en regel med denne templaten.

4.5.2.3 Security incident rules.

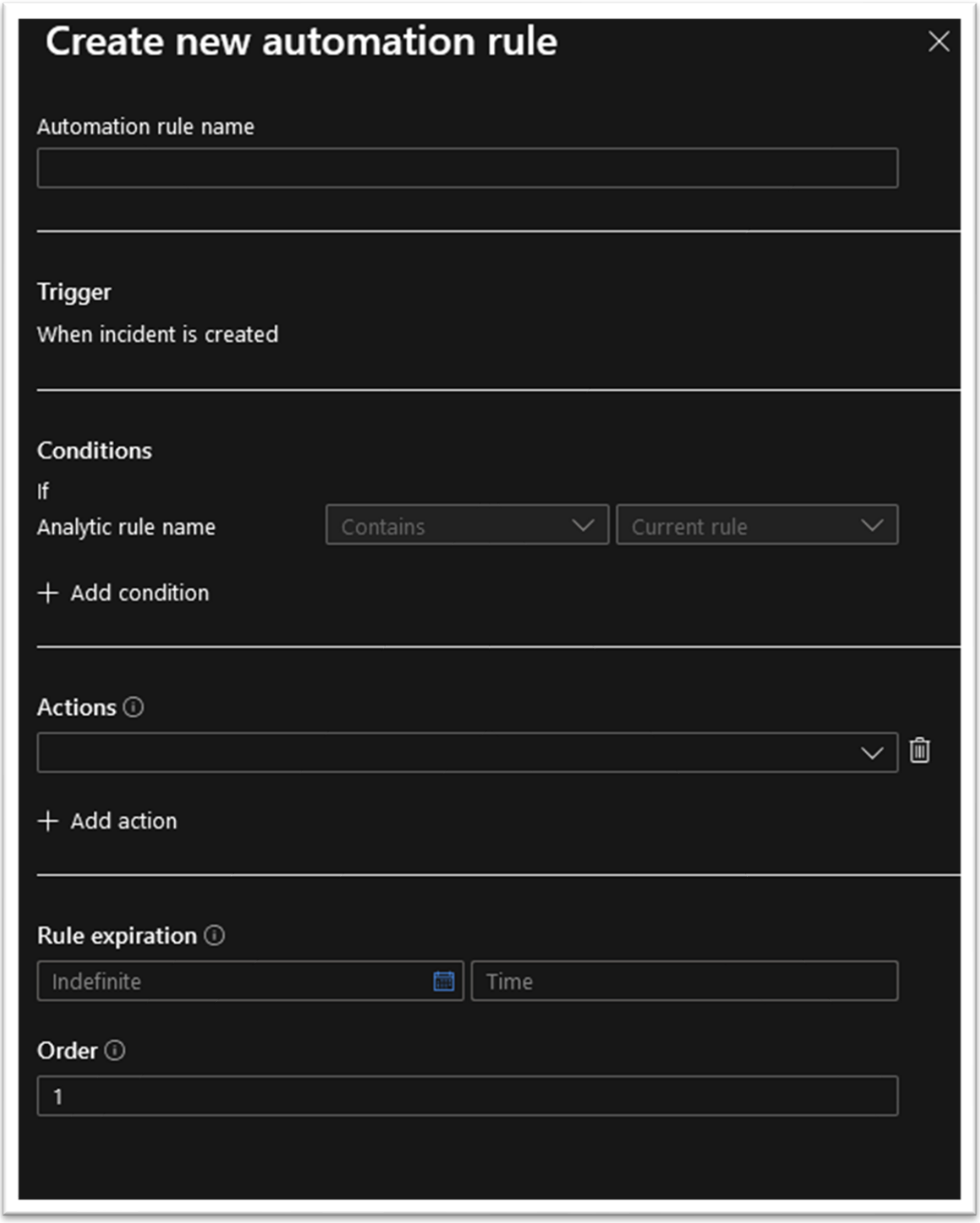
Her kan det velges hvilke Microsoft security løsning man vil hente informasjon fra, og man har ulike valg.



Man skriver et navn og en beskrivelse, velger hvilke Microsoft Security service man ønsker å basere regelen på, velger alvorlighetsgraden i filteret. Så kan man velge spesifikke alerts som

for eksempel «Failed signins» og man kan også utelukke spesifikke alerts dersom det er noen man ikke ønsker.

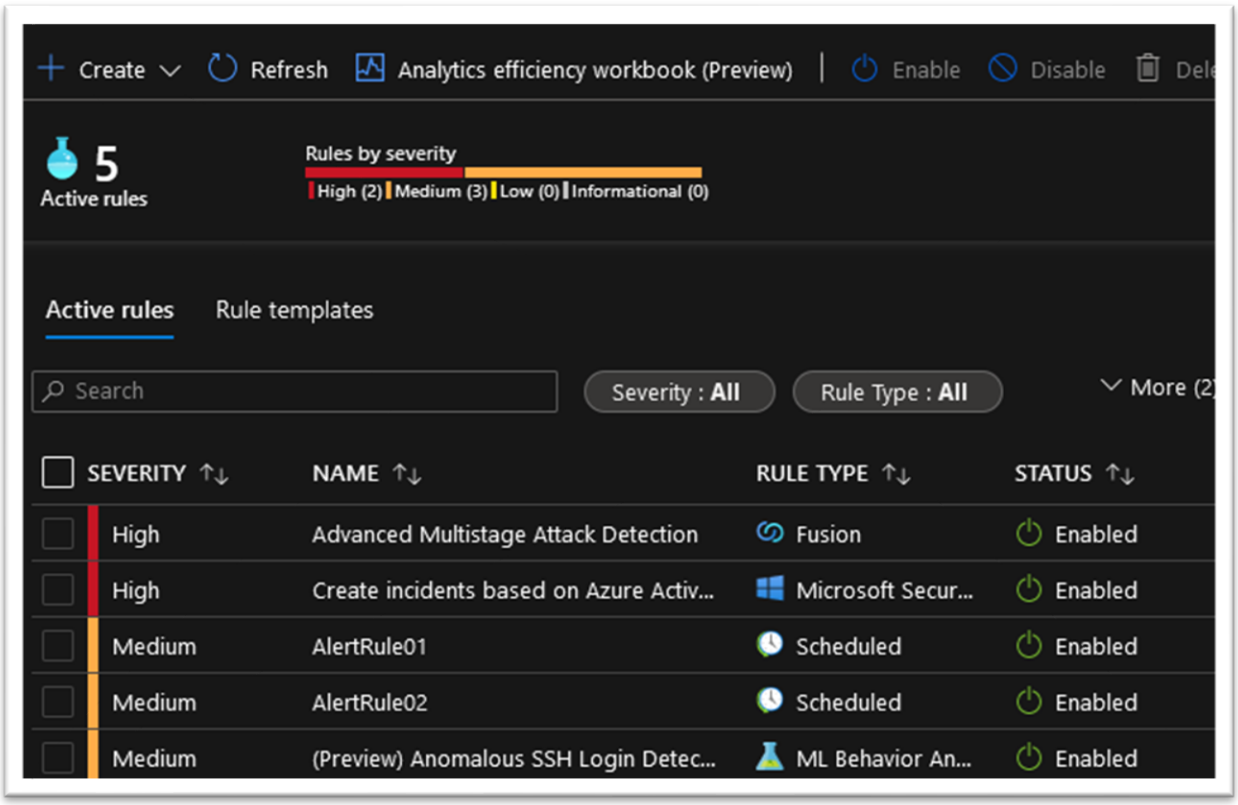
Videre konfigurerer man den automatiske responsen i en automation rule.



Der man gir den et navn, kan definere en trigger og respons man ønsker dersom triggeren oppfyller kravene som er satt i regelen.

4.5.2.4 Maskinlæring og adferdsanalyse

Disse templatene er basert på maskinlæringsalgoritmer så man kan ikke se eller endre den interne logikken for hvordan de kjører, men det finnes flere templates man kan lage regler ut fra når det kommer til denne typen regler og man kan derfor dekke et vidt spekter med forskjellige regler her.



Eksempel på alert rules

```
let timeframe = ago(3h);

let threshold = 2;

Okta_CL

| where published_t >= timeframe

| where eventType_s =~ "user.session.start"

| where outcome_result_s =~ "SUCCESS"

| summarize StartTime = min(TimeGenerated), EndTime = max(TimeGenerated)
```

En bruker logger inn fra forskjellige land med 3 timers mellomrom eller mindre. Dette er en enkel, men effektiv regel som hindrer en bruker fra å logge på sin bruker i organisasjonen fra flere land innen 3 timer.

```
# Azure DevOps Agent pool laget og slettet

let lookback = 14d;

let timewindow = 7d;

AzureDevOpsAuditing

| where TimeGenerated > ago(lookback)

| where OperationName =~ "Library.AgentPoolCreated"

| extend AgentCloudId = tostring(Data.AgentCloudId)

| extend PoolType = iif(isnotempty(AgentCloudId), "Azure VMs", "Self Hosted")

// Comment this line out to include cloud pools as well

| where PoolType == "Self Hosted"

| extend AgentPoolName = tostring(Data.AgentPoolName)

| extend AgentPoolId = tostring(Data.AgentPoolId)

| extend IsHosted = tostring(Data.IsHosted)

| extend IsLegacy = tostring(Data.IsLegacy)

| extend timekey = bin(TimeGenerated, timewindow)

// Join only with pools deleted in the same window
```

Siden vi bruker Azure dev ops er det viktig for oss å sikre at ingen uatoriserte har tilgang. Denne regelen hjelper oss å sørge for at det ikke skjer noen mistenkelig aktivitet på vår devops ved at den ser på «pools» som lages og slettes igjen innen 7 dager.

4.5.2.5 Deployment av rules fra DevOps

Når reglene er blitt opprettet og validert i Sentinel kan vi flytte dem over i DevOps, og så bruke pipelines for å ta i bruk reglene i andre workspaces. Prosessen for dette er ganske lik de andre pipelinene vi har brukt til nå. Før vi starter må vi redigere pipeline-filen AnalyticsRulesCICD.yml til å referere til riktig variabelgruppe og resource manager. Henholdsvis linje 48 og 57.

```
41 - stage: deploy_alert_rules
42   jobs:
43     - job: AgentJob
44       pool:
45        name: Azure Pipelines
46        vmImage: 'windows-2019'
47       variables:
48       - group: Az connection settings
49       steps:
50       - download: current
51         artifact: RulesFile
52       - download: Scripts
53         patterns: '*.ps1'
54       - task: AzurePowerShell@4
55         displayName: 'Create and Update Alert Rules'
56         inputs:
57          azureSubscription: 'AzureRG'
58          ScriptPath: '$(Pipeline.Workspace)/Scripts/Scripts/CreateAnalyticsRules.ps1'
59          ScriptArguments: '-Workspace $(Workspace) -RulesFile analytics-rules.json'
60          azurePowerShellVersion: LatestVersion
```

Når dette er gjort kan vi begynne å bygge analytics-rules.json. Siden det finnes fire ulike typer analytics rules i Sentinel, må vi sørge for at formatet på filen stemmer i forhold til det scriptet forventer. Scriptet for å deploye analytics rules til sentinel forventer følgende format.

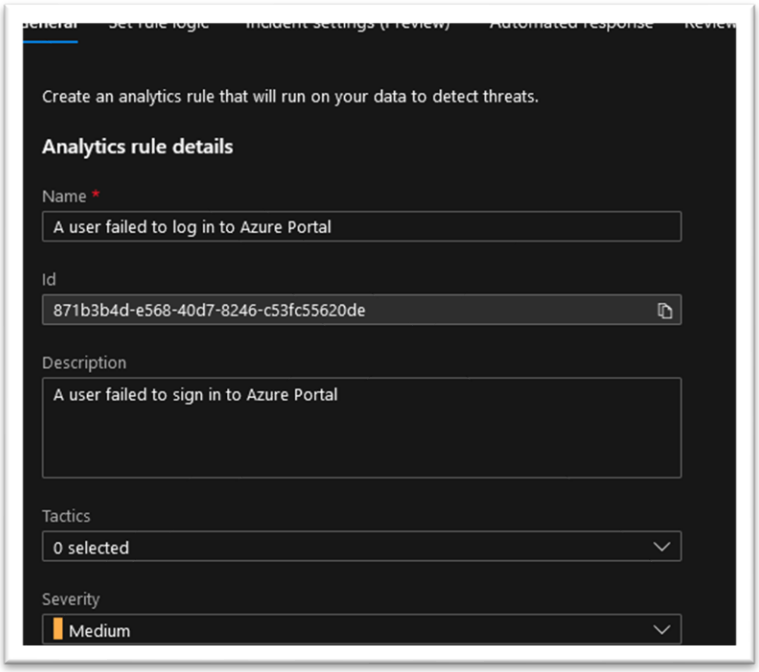
Fremgangsmåten for å hente ut listen over rules er den samme som vi gjorde med connectors tidligere. Denne gangen bruker vi følgende kommando, fortsatt med ConvertTo-Json.

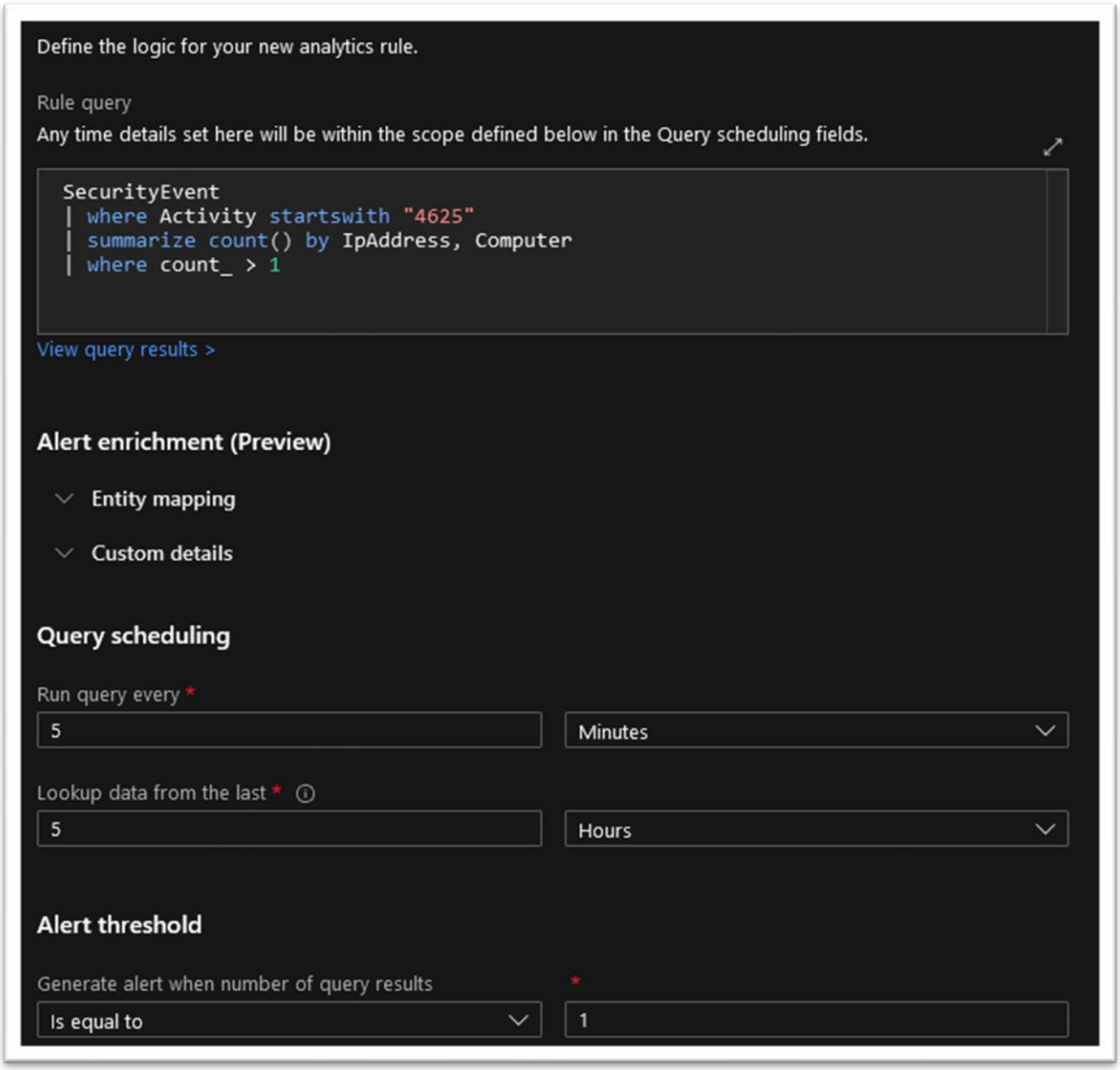
Last ned filen og legg inn hver regel under riktig topptekst. Dersom du har kjørt analytics rules-pipelinen tidligere, vil den automatisk bli startet av at vi gjør endringer til alert rules, dette skjer fordi det ligger en trigger i pipeline-filen.

4.6 Skrive og aktivere playbooks

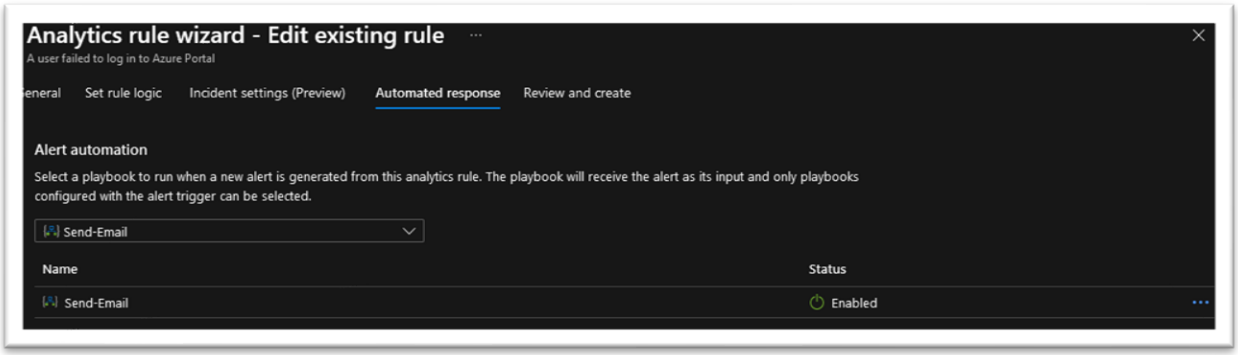
4.6.1 Playbooks i Sentinel

Først må vi lage en regel, her lager vi en enkel regel for å sjekke om en bruker mislykkes i å logge på Azure portal tjenesten. Vi setter denne regelen til medium alvorlighetsgrad for å sjekke at playbooken responderer slik den skal.

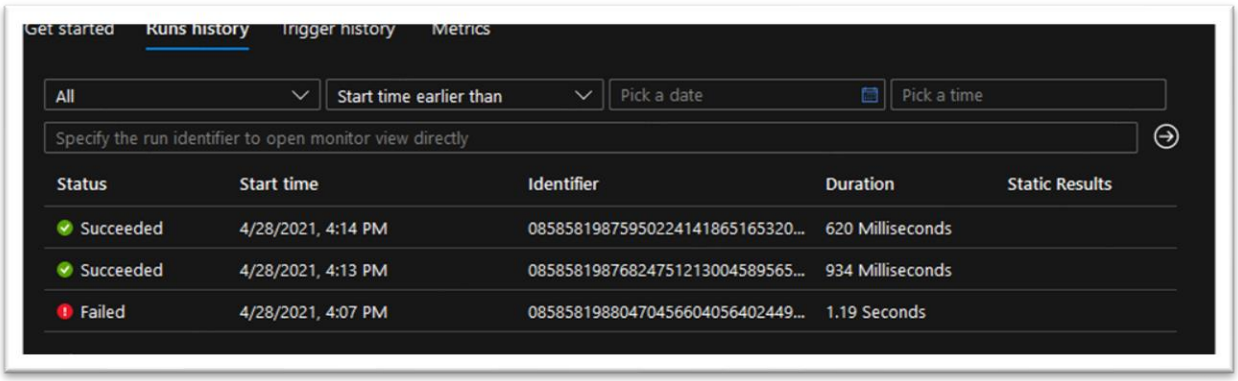

Create an analytics rule that will run on your data to detect threats.
Analytics rule details
Name *
A user failed to log in to Azure Portal
Id
871b3b4d-e568-40d7-8246-c53fc55620de
Description
A user failed to sign in to Azure Portal
Tactics
0 selected
Severity
Medium



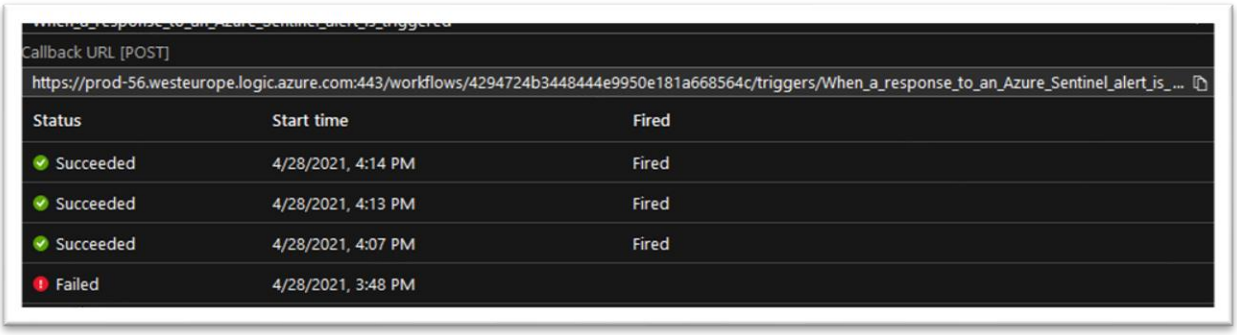
Her ser vi spørringen der security eventen som starter med «4625» er IDen til hendelsen mislykkede innlogginger. Vi ser også at spørringen kjøres hvert 5 minutt og sjekker data for de siste 5 timene. Vi setter også en grense for hvor mange slike hendelser som skal til for når alerten skal opprette en incident, som betyr at Sentinel da enten varsler en administrator eller kjører en playbook for å håndtere hendelsen. (Denne er bare satt til 1 for testing og anbefales ikke i et virkelig miljø der det skjer feil stadig vekk uten at det er noe alvorlig).



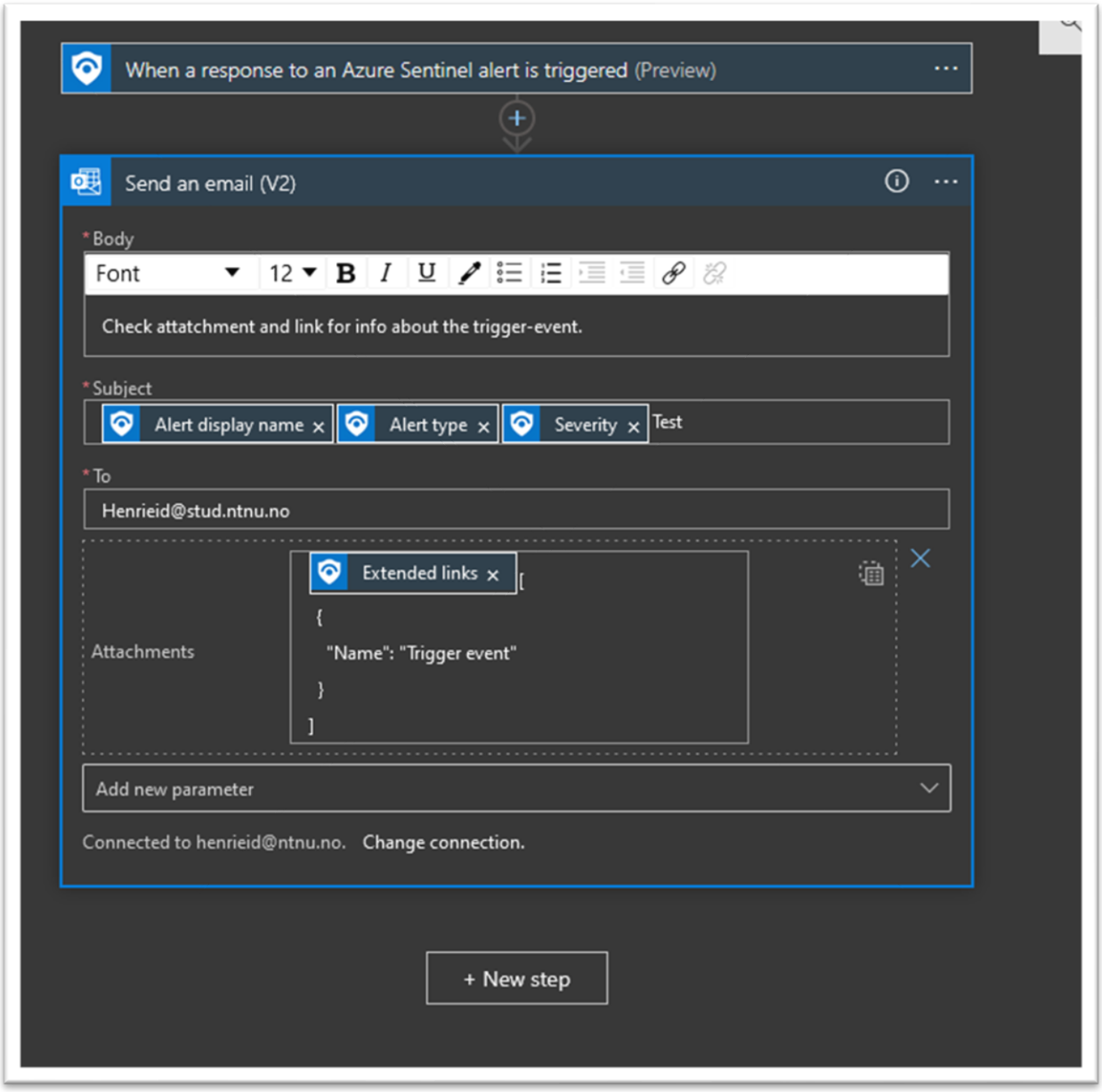
Her ber vi playbooken «Send email» respondere på alert rulen når den trigges, og det vil da skje en automated response umiddelbart etter at regelen er trigget i sentinel.



Vi ser her at playbookens første run feilet fordi det var en feilkonfigurering, men at den senere ble rettet opp og at den kjørte som den skulle i neste forsøk, før den i tredje forsøk ble trigget av en faktisk hendelse.



Her ser vi en callback URL som viser hva som har trigget hendelsen samt en log på når og hvordan det skjedde, samt om det var en suksess eller ikke.



Her ser vi hvordan playbooken «Send E-mail» er bygget opp. Den begynner altså med steget «When a response to Azure Sentinel alert is triggered» Og det er jo akkurat dette vi ønsker at når en alarm blir trigget så skal den reagere.

Det den gjør når den reagerer er at den lager en epost med emne som inneholder navnet på alerten som har blitt trigget, hva slags type alert det er og hvor viktig den er, slik at de som skal respondere og hvor viktig det er at den behandles.

Denne sendes altså til de epost adressene man ønsker, det kan være flere, det kan være grupper, eller Microsoft teams grupper. Det som sendes her er extended links som gir data og

informasjon om hva som har skjedd og hvordan i et vedlegg. Samtidig sendes beskjeden «Check attatchments for info about trigger event»

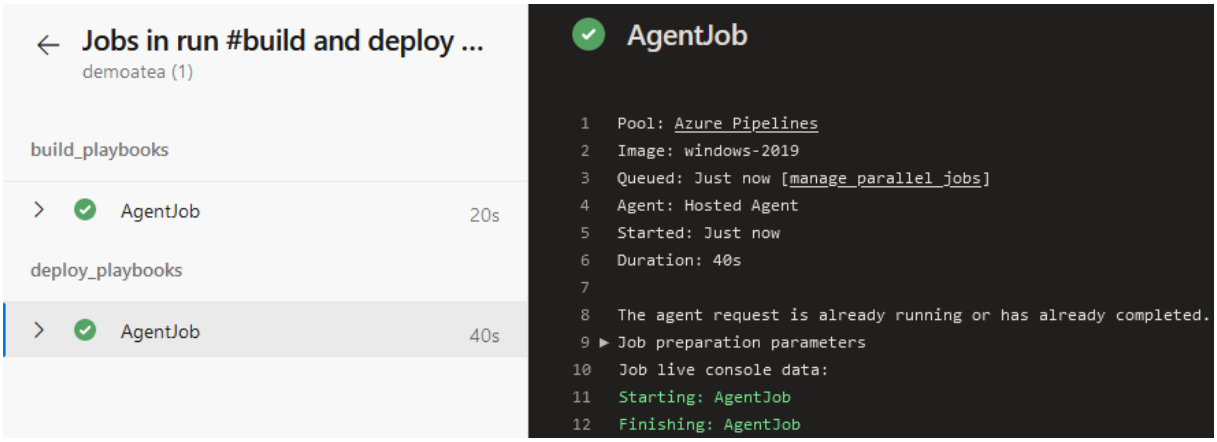
Ideelt sett skal denne sendes fra en dedikert konto.

4.6.2 Playbooks i DevOps

Deployment av playbooks i DevOps fungerer på samme måte som analytics rules. Vi endrer pipeline-filen playbooksCICI.yml til å referere til riktig resource manager og variabelgruppe. Henholdsvis linje 60 og 51.

```
50      variables:
51      - group: Az connection settings
52      steps:
53      - download: current
54        artifact: Playbooks
55      - download: Scripts
56        patterns: '*.ps1'
57      - task: AzurePowerShell@4
58        displayName: 'Create and Update Playbooks'
59        inputs:
60          azureSubscription: 'AzureRG'
```

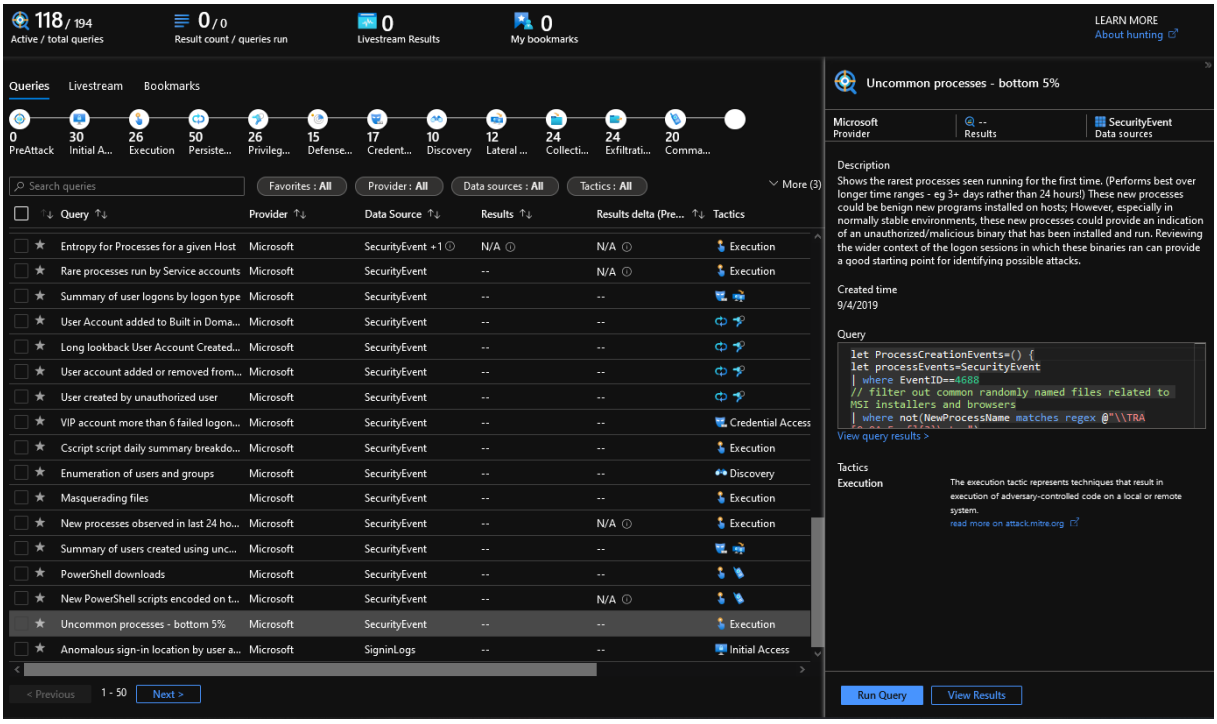
Deretter kan vi opprette og kjøre pipelinen.



Som med analytics rules blir de nye reglene hentet fra playbooks.json.

4.7 Skrive og aktivere hunting rules

4.7.1 Hunting rules i Sentinel



Hunting er et kraftig verktøy for å drive mer avansert sikkerhetsarbeid med Sentinel. Hunting skiller seg fra analytics rules i det at de ikke kjøres i bestemte intervaller, men må kjøres og analyseres manuelt. Hunting rules hjelper deg å sortere gjennom den enorme mengden data som kan bli sendt til Sentinel, og lar deg proaktivt lete etter uregelmessigheter som ikke blir

oppdaget av analytics, eller hendelser som du ikke nødvendigvis vil bli varslet om hver gang de skjer.

Regelen vi har markert i skjermbildet over viser en oversikt over de prosessene som blir minst brukt. Loggene for dette kommer fra alle maskiner som er koblet til Azure AD og er noe en sikkerhetsekspert kan undersøke etter mistenkelig aktivitet.

Du kommer svært langt med de innebygde reglene, men det er enkelt å skrive egne regler med KQL.

4.7.2 Hunting rules i DevOps

Deployment av hunting rules i DevOps fungerer på samme måte som analytics rules. Vi endrer pipeline-filen huntingRulesCICI.yml til å referere til riktig resource manager og variabelgruppe. Henholdsvis linje 60 og 51.

```
49      variables:
50      - group: Az connection settings
51      steps:
52      - download: current
53        artifact: HuntingFile
54      - download: Scripts
55        patterns: '*.ps1'
56      - task: AzurePowerShell@4
57        displayName: 'Create and Update Hunting Rules'
58        inputs:
59          azureSubscription: 'AzureRG'
60          ScriptPath: '$(Pipeline.Workspace)/Scripts/Scripts/CreateHuntingRulesAPI.ps1
61          ScriptArguments: '-Workspace $(Workspace) -RulesFile hunting-rules.json'
62          azurePowerShellVersion: LatestVersion
```

4.8 Fremtidig arbeid

Siden dette arbeidet er et «proof-of-concept» har vi oppdaget at er det mye videre arbeid som er mulig. Dette prosjektet kan karakteriseres som et «minimum viable product» og har rom for videreutvikling. Og med tanke på at oppdragsgiver kanskje er interessert i å selge Sentinel-løsninger som produkt til kundene sine, har vi vi følgende forslag for videre arbeid.

1. Automatisk hente rules fra en dev workspace Å opprette et workspace som blir brukt for å teste nye regler vil la ansatte utforske mulighetene som ligger i Sentinel videre uten at det påvirker et aktivt Sentinel workspace.
2. Workbooks og notebooks Workbooks og notebooks er nyttige verktøy som vi har sett lite på i dette prosjektet.
3. MSSP tenant for å beskytte IP. MSSP tenant er en spesiell tenant som kan opprettes under en subscription, og gjør at Atea kan aktivere Sentinel for en kunde, uten at de får tilgang til regler og konfigurasjon som er brukt mye tid på å utvikle. Queries, playbooks og workbooks kan overføres mellom tenants, med bruk av Lighthouse. Rules er planlagt.

Referanser:

Wortell, AZSentinel, 2020, tilgjengelig fra https://github.com/wortell/AZSentinel

Vårt arbeid med DevOps baserer seg i stor grad på kode og artikler publisert av Javier Soriano og Pouyan Khabazi. I tillegg finnes det et stort miljø av frivillige som deler regler og informasjon på den offisielle Sentinel git repoen.

Azure-Sentinel git, tilgjengelig fra https://github.com/Azure/Azure-Sentinel

Pouyan Khabazi, 2020:

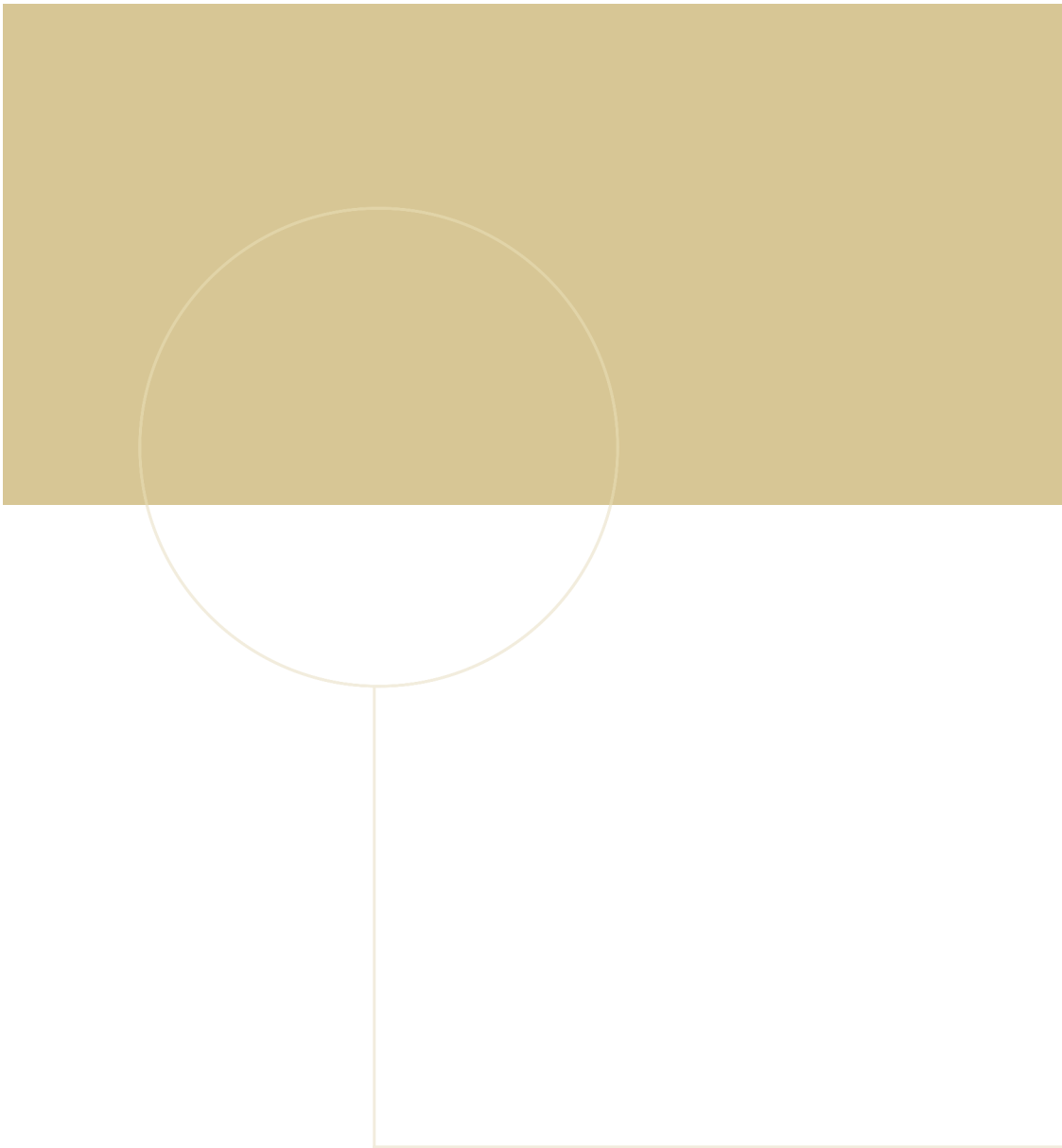
https://github.com/pkhabazi/sentineldevops

https://pkm-technology.com/deploying-and-managing-azure-sentinel-ninja-style/

Javier Soriano, 2020:

https://github.com/javiersoriano/sentinelascode

https://techcommunity.microsoft.com/t5/azure-sentinel/deploying-and-managing-azure-sentinel-as-code/ba-p/1131928



NTNU

Kunnskap for en bedre verden