UNIVERSITETET I AGDER

IS-304 våren 2022

Tittel: Sikker datahåndtering ved integrering av backuptjeneste

Emnekode	IS-304
Emnenavn	Bacheloroppgave i informasjonssystemer
Emneansvarlig:	Hallgeir Nilsen
Veileder	Janis Gailis
Oppdragsgiver:	Netsecurity

Studenter

Etternavn	**Fornavn**
Williams Antonsen	Elina
Erland	Kjartan
Larsen	Hans Christian

Jeg/vi bekrefter at vi ikke siterer eller på annen måte bruker andres arbeider uten at dette er oppgitt, og at alle referanser er oppgitt i litteraturlisten.	JA X	NEI___
Kan besvarelsen brukes til undervisningsformål?	JA X	NEI ___
Vi bekrefter at alle i gruppa har bidratt til besvarelsen	JA X	NEI___

Forord

Denne rapporten er skrevet for å dokumentere prosjektets prosess og for personer som har rede på de tekniske fagbegrepene innenfor systemutvikling.

Vi vil benytte denne muligheten til å takke alle de som har bidratt til vårt bachelorprosjekt. Det har vært en utfordrende, men lærerik prosess hvor kunnskap og råd fra bidragsyterne har bidratt i stor grad.

Takk til Janis Gailis for god veiledning gjennom prosjektet. Du har gitt oss gode tilbakemeldinger knyttet til innhold i rapport og prosjektstyringen. Du har også alltid vært tilgjengelig de gangene vi har hatt behov for veiledning og tilbakemelding.

Takk til Frank Kirkeng fra Netsecurity avdeling Kristiansand. Du var åpen for muligheten til å jobbe med dette prosjektet hos dere og vi har siden dag én blitt tatt godt vare på hos dere.

Takk til Cato Robstad som også har vært til god hjelp gjennom hele prosjektperioden hos Netsecurity. Det har vært til stor hjelp å ha deg som veileder da du satt på mye kunnskap som passet godt til dette prosjektet.

Til slutt vil vi også takke Martin Langballe for god hjelp og gode tilbakemeldinger relatert til det mest tekniske i integrasjonen.

Kjartan Erland

Elina W. Antonsen

Hans Christian Larsen

Sammendrag

Denne rapporten er skrevet i tilknytning til bacheloroppgaven i emnet IS-304 våren 2022. Rapporten er skrevet av BlueTeam, som er en gruppe bestående av tre studenter ved Universitet i Agder. Oppgaven er skrevet i samarbeid med bedriften Netsecurity, og går ut på å lage et nytt tjenesteelement til deres orkestreringsplattform ved bruk av Rubrik sin backup løsning.

Resultatet av prosjektet er et tjenesteelement som oppretter sikkerhetshendelser i orkestreringsplattformen når en feil oppstår i Rubrik sin backuptjeneste. Metadataen som trengs for å opprette en slik incident blir hentet ved bruk av en GraphQL-API. For å teste at tjenesteelementet funket i sin helhet med alle nødvendige funksjoner så ble det gjennomført en akseptansetest av en ansatt i bedriften. Gruppen fikk dermed en tilbakemelding på hvordan opplevelsen av tjenesten var.

Scrumban rammeverket har blitt brukt til å sikre prosjektstyring og agilitet gjennom hele prosjektet. Scrumban er en metodikk som er mer fleksibel enn scrum, som er det rammeverket som ellers har blitt brukt gjennom studieløpet. Ettersom at gruppen skulle begi seg ut på noe de ikke hadde vært borti før og utfordringer kunne oppstå, var det et bevisst valg å benytte Scrumban. På denne måten kunne gruppen håndtere eventuelle utfordringer samtidig som de kunne sikre kontinuerlig fremgang i prosjektet. I tillegg så hadde ikke bedriften noen rammeverk de praktiserte og gruppen sto fritt til å velge det som passet dem.

Gruppen har fra begynnelsen av prosjektet vekslet mellom digitalt og fysisk arbeid sammen. Dette har vist seg å være en god måte å håndtere arbeidet på og det møter alle behovene til gruppens medlemmer. Det har vært viktig å ha en rutine for arbeidet og gruppen satte tidlig et tidspunkt for kjernetiden de ønsket å arbeide sammen. Discord har vært et viktig verktøy for gruppen, de har brukt dette til å kommunisere seg imellom ved de digitale gruppetimene. Microsoft To Do og Miro er de to andre verktøyene som har vært svært viktig i gruppens håndtering av oppgaver og gjøremål. Disse verktøyene har bidratt til oversikt og kontroll gjennom hele prosjektet.

Tjenesteelementet ble testet av personer utenfor gruppen slik at de kunne få tilbakemeldinger og et annet syn på tjenesten. Resultatene som kom frem som var som gruppen forventet. I forkant av testen hadde testperson fått lest akseptansekravene og tilbakemeldingen var at han syntes tjenesteelementet fungerte i samsvar med hva som var forventet. Selv om tjenesteelementer oppfyller de viktigste punktene så er det lagt opp til at den skal kunne videreutvikles ettersom hvilke oppdateringer som kommer fra Rubrik.

Innholdsfortegnelse

Figurliste

1. Introduksjon

BlueTeam er en gruppe på 3 studenter som ble etablert sent i høsten 2021. Deltakerne hadde ikke noe særlig kjennskap til hverandre fra før, så bakgrunnen for opprettelsen av gruppen var en felles interesse for sikkerhet og at de hadde hatt praksis sammen i bedrift.

Allerede før gruppen ble offisielt dannet snakket studentene om forventninger og motivasjon for å gjennomføre en bacheloroppgave sammen. Da det viste seg at gruppemedlemmene hadde like ambisjoner, ble det bestemt at de skulle skrive sammen. Høsten 2021 hadde studentene praksis hos Netsecurity og luftet allerede da muligheten for å ha bacheloroppgaven hos dem. Det viste seg at bedriften hadde flere alternativer og ga gruppen mulighet til å velge den oppgaven de måtte ønske. Hovedgrunnen for at gruppen etterspurte et samarbeid med Netsecurity var interessen for sikkerhet og et ønske om å skrive om noe sikkerhetsrelatert. Oppgaven gruppen valgte er for bedriften rettet mer mot deres drift av sikkerhetstjenester enn sikkerhet i seg selv, men for gruppen virket det svært interessant å være med på å utvikle noe som vil være tilnærmet revolusjonerende for backup-tjenester.

I løpet av dette semesteret har gruppen jobbet kontinuerlig med prosjektet, og hatt god kommunikasjon innad i gruppen samt med oppdragsgiver har vært nødvendig for å sikre fremgang. Det som har vært spesielt interessant er Netsecurity sin satsning på nye tjenester og fokus på innovasjon.

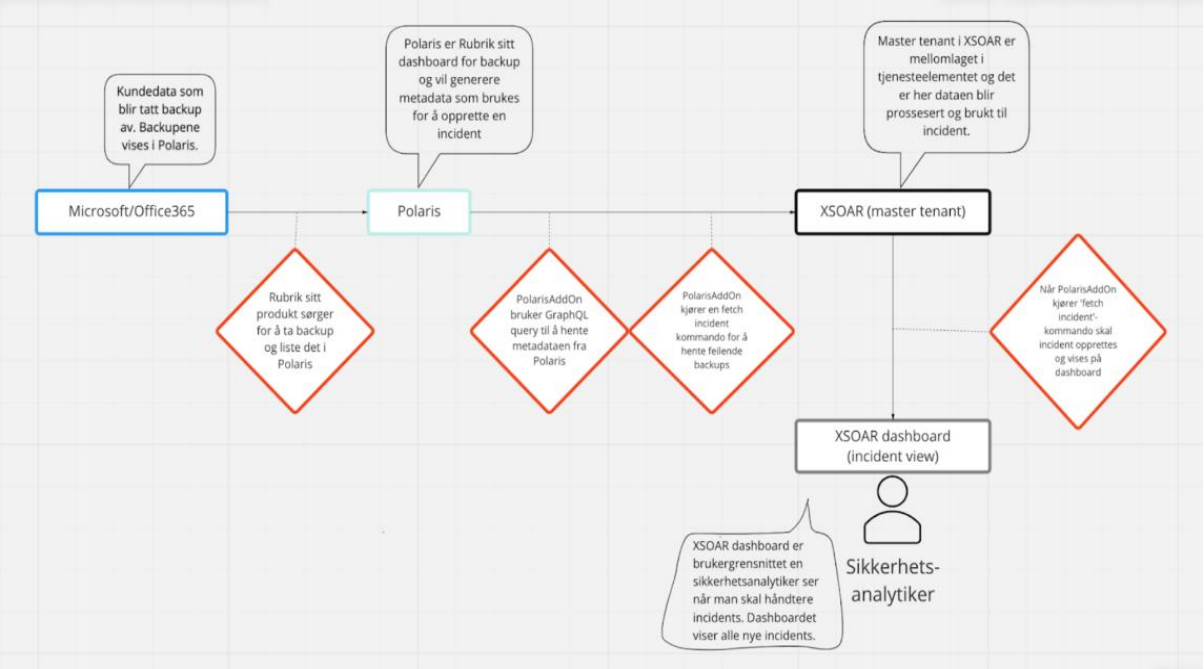
Prosjektet går ut på å lage et nytt tjenesteelement som skal automatisere onboarding, leveranse og oppfølging av O365 og Rubrik Air kunder i størst mulig grad. Tjenesteelementet skal ved hjelp av GraphQL-API hente metadata fra Rubrik sin backuptjeneste og ut ifra denne informasjonen opprette en incident i orkestreringsverktøyet til Netsecurity slik at sikkerhetsanalytikere kan håndtere hendelsen med den informasjonen som er tilgjengelig i incidenten.

2. Tjenesten

I dette prosjektet har gruppen laget et nytt tjenesteelement til bedriftens orkestreringsverktøy. Arbeidet har gått ut på å ferdigstille er en løsning hvor bedriftens sikkerhetsanalytikere enkelt kan håndtere incidents knyttet til sikker datahåndtering. Incidents blir hentet fra Rubrik Polaris som er et dashbord for backup, og informasjonen fra dashbordet lastes automatisk inn med all data som er nødvendig for å gjøre en vurdering av incidenten. Målet var dermed at sikkerhetskopier av dataene generert av Netsecurity sine kunder i ulike skyløsninger som Office, Teams og Exchange, skal beskyttes og overvåkes av Rubrik, med alarmer som går til mennesker kun når systemet ikke kan ta en beslutning selv.

Tjenesteelementet er delt opp i en trelagsarkitektur (IBM, 2020), fordelt over to systemer. Det første datalaget består av Rubrik sitt Polaris-grensesnitt, som brukes til fremvisning av backup. GraphQL-API brukes for å hente metadata om backups inn til Palo Alto XSOAR (Security Orchestration Automation Response), som er sikkerhetsorkestreringsplattformen til Netsecurity. Det andre logiske laget er XSOAR «master tenant» som prosesserer data fra datalaget og som skal videre til presentasjonslaget. I dette laget er det blitt skrevet en integrasjon i Python, som er systemets måte å hente inn og konvertere data fra andre kilder på.

Videre er det blitt satt opp en såkalt “playbook”, en samling av skript og funksjoner i et grafisk programmerbart flytdiagram som brukes for å håndtere ulike typer incidents. Se skjermbilde 2 i kapittel 3.3 for nærmere beskrivelse. Denne integrasjonen, i samarbeid med playbook, blir satt på enten “Master” eller “Kunde”-siden, eller tenant som det heter i XSOAR. Herfra blir incidents opprettet i XSOAR dashboard som er presentasjonslaget i arkitekturen. I presentasjonslaget blir incidents presentert i brukergrensesnittet analytikerne bruker for å håndtere incidents. Figur 1 på neste side er en illustrasjon som viser flyten til tjenesteelementet samt hva de forskjellige elementene gjør.



Figur 1: Flyten og elementene til tjenesteelementet.

3. Sentrale avgjørelser

I denne delen vil de mest sentrale avgjørelsene i prosjektet rundt teknologier, prosjektstyringsverktøy og kommunikasjonsplattform drøftes. I hver av nevnte delene vil det gjennomgås ytterlige emner rundt disse.

3.1 Vinkling av prosjektoppgave

I starten ønsket gruppen å jobbe med noe sikkerhetsorientert. Gruppen ble dannet i løpet av praksisperioden høsten 2021 som også foregikk hos Netsecurity. Mot slutten av denne perioden var gruppen i flere møter med mulig oppdragsgiver for å lage en prosjektoppgave.

Det ble raskt valgt å arbeide med en sikkerhetsorientert løsning med backup. Netsecurity, i samarbeid med Rubrik, som er et skylagring-selskap med fokus på sikker datahåndtering ved å motarbeide f.eks. løsepengevirus, ønsket å få implementert sikker datahåndtering inn til Netsecurity sitt orkestreringsverktøy XSOAR (Rubrik, 2022).

En stor utfordring i utgangspunktet var å skaffe seg nok kompetanse og kunnskap til å komme i gang med prosjektet. Ettersom denne backuptjenesten fra Rubrik er en relativt ny tjeneste, og implementasjonen mot Netsecurity er ny, så var en del kunnskapsheving nødvendig å gjennomføre. Her startet prosessen med å ta Rubrik Technical Associate-kurset for å få en grunnleggende forståelse av teknologien bak prosjektoppgaven. Valget om å gjennomføre dette kurset var grunnet i et ønske om å få en forståelse for hva Rubrik er og hva deres tjenester kan bidra med til prosjektet.

Tidlig i oppstartsfasen av prosjektet ble det klargjort fra oppdragsgiver at oppgaven skulle fokusere mer på en mer driftsorientert løsning, ikke sikkerhetsorientert. Dette var fordi Netsecurity hadde “backup” som sitt hovedmål i første omgang, hvor sikkerhetsløsningen mot løsepengevirus var en tilleggsfunksjon som skulle komme i etterkant.

3.2 Programmeringsspråk

Det var tydelig allerede fra dag én at det skulle skrives en integrasjon i orkestreringsplattformen til Netsecurity, hvor backup-data skulle håndteres. Alle integrasjoner på denne plattformen er skrevet i Python, og gruppen hadde dermed ikke noe valg i hva slags programmeringsspråk som skulle brukes. Heldigvis finnes det retningslinjer for hvordan integrasjoner skrives på orkestreringsplattformen, ettersom de følger en viss standard. Disse retningslinjene gjorde arbeidet til gruppen enklere å gjennomføre. Ettersom Python er relativt enkelt å jobbe med var de største utfordringene å knytte Python-koden i integrasjonen sammen med API-et, noe som blir nærmere diskutert i kapittel 3.4.

Ifølge Python sine hjemmesider er fokuset til programmeringsspråket funksjonsbasert, og "lar deg jobbe kjapt med å integrere systemer effektivt" (Python.org). Kombinert med at orkestreringsplattformen er bygget opp av relativt korte programmer med spesifikke funksjoner, så er Python et utmerket valg.

I samtale med en av Netsecurity sine sikkerhetsanalytikere ble det nevnt at Python har vært et bra valg av programmeringsspråk til formålet. Lesbarhet og hvor lett det er å skrive ble trukket frem som eksempler. Enda en fordel med dette var at de nye i jobb raskt er i gang med å skrive produktiv kode i XSOAR. Gruppen er av den oppfatning at det generelt sett har vært en positiv opplevelse å bruke Python, selv om det til tider har vært utfordringer knyttet til å bruke det som i starten var et helt fremmed programmeringsspråk.

3.3 Orkestreringsverktøy

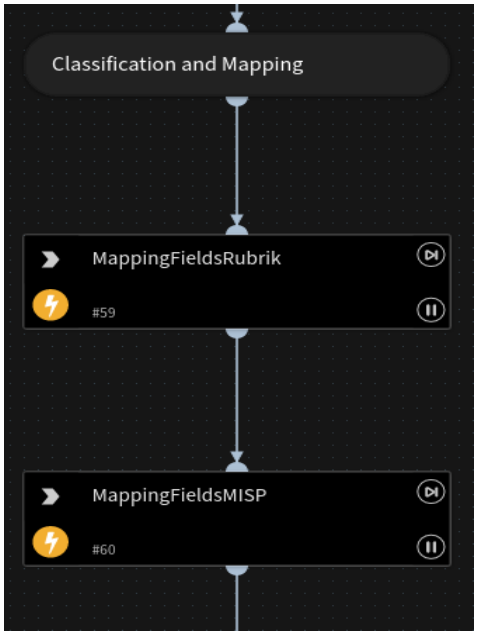
Hele produksjonen til Netsecurity er avhengig av et orkestreringsverktøy fra Palo Alto som heter XSOAR. Dette er et sikkerhetsorkestrerings-, automasjon- og responsplattform som håndterer alt av saksbehandling, automasjon og sikkerhetshåndtering (Palo Alto Networks, u.å). Dette verktøyet er et knytningspunkt for alle Netsecurity sine operasjoner, og brukes ved at man skriver integrasjoner i Python, og henter inn informasjon ved å hovedsakelig bruke ulike API-er. Dette verktøyet har et stort utviklingspotensial spesielt når det kommer til å ta i bruk API-er fra andre tjenester.

XSOAR består av to hoveddeler; integrasjoner og playbooks. Playbooks er motoren i det hele (Cortex XSOAR, 2020), og er måten man automatiserer integrasjoner. Integrasjonen sender spørringer og brukes for å manipulere dataene til et format som XSOAR forstår. I gruppens prosjektoppgave skal det hentes hendelsesdata fra backup, noe som vil si at en integrasjon i XSOAR bruker en funksjon for å spørre etter data, og playbooks vil automatisere hentingen av dette, og til slutt lage en incident i XSOAR sitt grensesnitt. Skjermbilde 1 under viser et utdrag av koden i en integrasjon skrevet i XSOAR sitt integrerte utviklingsmiljø.

```
    events = client.gql_query(query)

    for event in events["data"]["activitySeriesConnection"]["edges"]:
        for key, value in event["node"].items():
            if key == "lastActivityType":
                name = value
            if key == "lastUpdated":
                incident_created_time = dateparser.parse(value)
            if key == "activityConnection":
                name += f': {value["nodes"][0]["message"]}'

        if incident_created_time > last_fetch_time:
            incident = {
                'name': name,
                'occurred': incident_created_time.strftime(DATE_FORMAT),
                'rawJSON': json.dumps(event)
            }
            incidents.append(incident)
            latest_created_time = incident_created_time

    next_run = {'last_fetch_time': latest_created_time.strftime(DATE_FORMAT)}
    return next_run, incidents

def main():
    """
        PARSE AND VALIDATE INTEGRATION PARAMS
    """
    params = demisto.params()
    polaris_account = params.get('polaris_account', False)
    polaris_domain_name = "my.rubrik.com"
    polaris_base_url = f"https://{polaris_account}.{polaris_domain_name}/api"
```

Skjermbilde 1: Utdrag fra koden til integrasjonen.

I skjermbilde 2 kan man se et utsnitt av hvordan en playbook ser ut. Denne delen av playbooken er ‘Klassifisering og Tilordning’, som er dens hovedfunksjon, og tilordner data hentet fra Polaris til felt i incident som blir generert.



Skjermbilde 2: Utsnitt fra playbook.

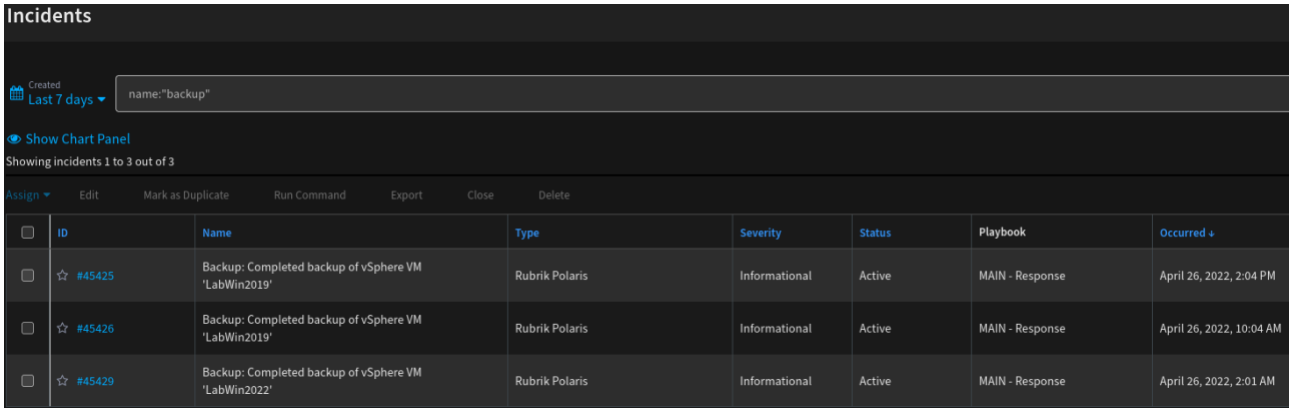
Som nevnt tidligere er playbooks automatisering av skript. Det vil si at “MappingFieldsRubrik” i playbook er et skript, som vist i skjermbilde 3 under, og er det som faktisk gjennomfører tilordning av data til felter i incidents.

MappingFieldsRubrik

```
65  # Parsing data
66  dArgs = {}
67  dArgs['rubrikcdmclusterid'] = node.get('cluster', '')['id']
68
69  sev_mapping = {
70      "Info": 0.5,
71      "Warning": 1,
72      "Low": 1,
73      "Medium": 2,
74      "High": 3,
75      "Critical": 4
76  }
77
78  if (sev := node.get('severity')) in sev_mapping:
79      dArgs['severity'] = sev_mapping[sev]
80
81  ### Incoming Alert Info
82  # Object
83  add_or_append(dArgs,'rubrikpolarisobjectname',node.get('objectName',''))
84  add_or_append(dArgs,'rubrikpolarisobjectid',node.get('id',''))
85  add_or_append(dArgs,'rubrikpolarisobjecttype',node.get('objectType',''))
86  # Cluster
87  #add_or_append(dArgs,'rubrikcdmclusterlocation',node.get('cluster', '')['location'])
88  add_or_append(dArgs,'rubrikcdmclustername',node.get('cluster', '')['name'])
89  add_or_append(dArgs,'rubrikpolarismessage',node.get('message',''))
90  add_or_append(dArgs,'rubrikpolarisactivityseriesid',node.get('activitySeriesId',''))
91  add_or_append(dArgs,'rubrikpolarislastactivitystatus',node.get('lastActivityStatus',''))
```

Skjermbilde 3: Script som blir kjørt i playbook.

I skjermbilde 4 under ser man en oversikt over hvor incidents blir generert. Dette skjer som et resultat av at en playbook automatisk kjører koden i integrasjonen, og oppretter en incident med relevant informasjon som alvorlighetsgrad, navn på server, med mer.



Skjermbilde 4: Grensesnitt som viser hvilke incidents som har blitt generert.

Gruppen snakket sammen med en sikkerhetsanalytiker ved Netsecurity som også hadde møtt på utfordringer med XSOAR. Blant disse var venting på grunn av tung Javascript som kjører på XSOAR sin nettside under bruk, i tillegg til to ulike flaskehalser i Docker, minneallokering i containere og å treffe grensen for maks antall samtidige containere når playbook-skript og integrasjoner blir kjørt. Minnebegrensningen for containere er der for å forhindre at en container tar for mye av systemets ressurser grunnet enten programfeil eller DoS-angrep, og er justerbar (Palo Alto Networks, 2022).

Det har også vært en rekke programfeil som har oppstått, men ettersom XSOAR-plattformen er åpen kildekode har det vært mulig for sikkerhetsanalytikerne å fikse feil som oppstår og lage pull requests for å hjelpe med å forbedre programmet for alle andre som bruker det óg.

Gruppen har hatt en del utfordringer med bruk av XSOAR som i hovedsak går på å ikke være godt nok kjent med plattformen. Grensesnittet bærer preg av å ha for mange knapper og undermenyer i flere lag, samt mye venting knyttet til kjøring av playbooks, skript og integrasjoner. I tillegg har gruppen også møtt feil og problemer som kun manifesterte seg i testmiljøet, noe som er grunnen til at man så seg nødt til å gjøre testing i produksjonsmiljø, istedenfor testmiljøet som ellers var brukt.

Det er viktig å nevne at dette testmiljøet var opprettet kun for gruppen å arbeide i. Utover dette er det ingen vesentlig forskjell enn at det er en annen konto man logger inn med gjennom nettsiden til XSOAR. Netsecurity har sin egen "Dev-tenant", dvs. en slags instans, eller side på XSOAR. Systemet bygger på arv, hvor alle kunders instanser arver funksjonaliteten til en "Master tenant", som er en instans der den generelle funksjonaliteten til XSOAR-plattformen er å finne. Hver kunde har også sin egen "tenant", så dersom en kunde skulle ønske å benytte seg av tjenesteelementet som gruppen har laget, skal det ikke være vanskeligere enn å synkronisere dette fra "Master tenant" til kundens egen tenant, noe som er tydelig i kartleggingsmodellen i Figur 1. Grunnen til dette er måten XSOAR er bygget opp med tanke på playbooks og integrasjoner, og hvordan de blir implementert i tenants.

3.4 Programmeringsgrensesnitt (API)

Gruppens prosjekt baseres på API fra Rubrik og GraphQL. Dette API-et brukes for å generere og hente metadata fra Rubriks grensesnitt Polaris. Via Polaris benyttes GraphQL for å hente metadata fra aktiviteten på alt av tjenester. GraphQL er et spørrespråk, et programmeringsspråk dedikert til API-er, og i motsetning til REST API-er som baserer seg på individuelle spørringer til elementer (GraphQL, u.å.). Ved å bruke GraphQL kan det lages spørringer i kode på orkestreringsverktøyet for å hente bulker med metadata, f.eks. navn på server en tar backup fra, status på backup osv. I skjermbilde 5 ser man spørringen som blir kjørt mot Polaris som henter metadata om backups som har feilet.

```
query = """query {
            activitySeriesConnection(first: 5, filters: {lastActivityStatus: Failure, lastActivityType: Backup}) {
              edges {
                node {
                  id
                  fid
                  activitySeriesId
                  lastUpdated
                  lastActivityType
                  lastActivityStatus
                  objectId
                  objectName
                  objectType
                  severity
                  progress
                  cluster {
                      id
                      name
                  }
                  activityConnection {
                      nodes {
                          id
                          message
                          severity
                          time
                      }
                  }
                }
              }
            }
          }
    """
```

Skjermbilde 5: Spørringen som blir kjørt mot Polaris.

API-et til Rubrik brukes for å sende metadata fra de forskjellige tjenestene i Polaris. Disse tjenestene er alt fra vanlig lagring av kundedata til automatiserte backup-typer og sikkerhetsorienterte løsninger som motarbeider løsepengevirus.

Bruksområdet for dette API-et er å se etter uventet aktivitet på en backup med kompromittert data, hvor XSOAR gjennom sine integrasjoner og playbooks skal kunne presentere denne alarmen slik at en sikkerhetsanalytiker kan håndtere situasjonen dersom systemet ikke selv klarer å håndtere alarmen.

Fordelen med å bruke GraphQL er at det virket å være veldig fleksibelt med spørringer, da man kan gjøre veldig konkrete spørringer for å hente den dataen du vil ha, istedenfor store bulkspørringer som REST-basert API gjør, som beskrevet av Sebastian Eschweiler i et innlegg på CodingTheSmartWay.com(Eschweiler, 2018). Med enkle endringer i spørringen kan man endre hvilken informasjon man henter og eventuelle filtre som benyttes.
Som man kan se i skjermbilde 5, er spørringen satt opp til å kun hente de fem første resultatene av typen "Backup" som feilet. Spørringene føres rett inn i integrasjonskoden i XSOAR, som gjør at Netsecurity enkelt kan gjøre endringer etter behov.

For kunne nyte fleksibiliteten og fordelene av GraphQL var det noen ulemper og konsekvenser som var viktige å ta hensyn til og reflektere over. Hovedutfordringen til gruppen innen dette området var å finne god dokumentasjon på både API-et til Rubrik og hvordan dette ble brukt med GraphQL. Gruppen klarte å finne fram deler av dokumentasjonen for å lage en grunnleggende spørring, men endte opp med å sende e-post til Rubrik og spørre om dokumentasjon direkte fra dem. Gruppen var heldige og fikk kjapp kontakt med Rubriks kontaktperson og løste denne problemstillingen relativt kjapt.

3.5 Prosjektstyringsverktøy

I dette prosjektet har ble det valgt å bruke Google Docs, et Excel-regneark for Gantt-diagrammet, og Microsoft To Do for å dele opp og delegere arbeidsoppgaver. Gruppen valgte å bruke Google Docs ettersom det er et program alle medlemmene hadde god erfaring med fra før. Microsoft To Do ble tatt i bruk fordi det virket som en god løsning for oss, hvor man enkelt kunne føre inn egne punkter som skal gjennomføres. Gantt-diagrammet som er tatt i bruk er hentet fra en mal (Vertex42, 2021).

I tillegg til dette brukte gruppen Miro som sitt hovedverktøy innenfor prosjektstyring, ettersom Miro lar deg simulere forskjellige prosjektstyringsrammeverk, som f.eks. Kanban.

I en kombinasjon med Microsoft To Do og Miro, dannet dette et godt grunnlag for gruppens prosjektstyring. Dette var et resultat av funksjonen til Microsoft To Do å ha en styringsmetode rettet mot det daglige og ukentlige.
Gruppen brukte dette som en grov mal og planlegger i korte perioder, for så bruke dette i Miro som har funksjoner dekket for prosjektstyringsrammeverk, som blir diskutert i kapittel 4.

Gruppen la tidlig merke til at XSOAR-plattformen ikke har noen nyttig form for versjonskontroll, da alt av kode blir skrevet rett inn i plattformen. Netsecurity bruker heller ikke en tradisjonell versjonskontroll som Github, eller systemutviklings-rammeverket DevOps. De bruker foreløpig heller Tasks funksjonen i Microsoft Teams. Denne funksjonen fungerer som et kanban-board hvor du kan sette opp detaljerte oppgaver til å fungere som en type versjonskontroll (Microsoft, u.å.). Her kan brukere sette opp oppgaver på hva som skal gjøres, innen tidsrammer og sette status på oppgaver. Denne løsningen førte til at gruppen måtte heller bruke kommunikasjon som et styringsverktøy til versjonskontroll. Det var derfor utrolig viktig å ha god kommunikasjon ofte med både veileder og gruppens kontaktperson på det tekniske.

3.6 Kommunikasjonsplattform

I hovedsak har gruppen benyttet seg av Discord for digital kommunikasjon og møter med hverandre. Gruppen bruker Discord fordi det er en plattform alle medlemmene har erfaring med fra før, i tillegg til at samtaler og diskusjoner lett kan foregå der. For møter med personer utenfor gruppen ble enten Teams eller Zoom tatt i bruk, avhengig av hvem møtet var med. Disse to programmene blir oftest brukt til møtevirksomhet og digital kommunikasjon, spesielt under pandemien for mange bedrifter og organisasjoner. I både Discord og Teams er det verktøy og funksjoner for ulike typer kommunikasjon som gruppen har tatt i bruk. På disse plattformene brukes det forskjellige kanaler med informasjonsdeling og kommunikasjon, sortert på f.eks. emner og avdelinger. Disse funksjonene gjør det enkelt og fleksibelt for et team med tre medlemmer å ha god struktur på kommunikasjonen.

4. Prosjektstyring

Prosjektstyring er en essensiell del av alle prosjekter og innebærer den kontinuerlige styringen av et prosjekt. Prosjektstyring skal ta for seg usikkerheter som oppstår i prosjektet, det skal sikre fremdrift samt håndtere planer og budsjett (Digdir, u.åa). Gruppen har ved hjelp av prosjektstyring hatt god oversikt over oppgaver som må gjøres og til hvilken tid disse burde være gjennomført for å holde seg etter estimert produktlevering.

4.1 Prosjektstyringsrammeverk

Det ble bestemt at gruppen skulle benytte seg av metoden scrumban. Dette kom av at gruppen ønsket et rammeverk som hadde noe av strukturen fra scrum og fleksibiliteten til kanban. Da oppdragsgiver ikke driver med utviklingsprosjekter i noen stor grad, hadde de derfor ingen innsigelser på valget av metodikk.

De fleste er kjent med prosjektmetodikkene Scrum og Kanban, derfor var det uforventet å komme over Scrumban som en etablert metode som blir brukt til prosjektarbeid. Scrumban gir som nevnt strukturen til scrum i form av bruken av sprinter og backlog samt ukentlige møter, mens kanban-delen gir arbeidsflyt ved at man ikke har noen bestemte tidsfrister, men heller et visst antall oppgaver påbegynt som må ferdigstilles før ny oppgave kan påbegynnes. (Kukhnavets, 2019).

Scrumban brukes av team som ønsker et enkelt rammeverk å implementere samtidig som det er tydelig og bruke. Det gir god oversikt over progresjon og de “korte” sprintene fører til motivasjon og mestringsfølelse da det virker overkommelig med slike intervaller (Chappell, 2020). Scrumban er det beste valget for gruppen ettersom man ønsker å se kontinuerlig progresjon og utnytte tiden godt. Dette passer godt inn med prosjektet ettersom de aller fleste oppgavene vil gi en form for resultat som kan bygges videre på.

4.2 Iterasjoner

Det er blant annet konseptet sprinter som scrumban har tatt med seg fra scrum metodikken, men i scrumban så kalles sprinter for iterasjoner og disse iterasjonene varer aldri lenger enn to uker (Chappell, 2020). For gruppen var det viktig å kunne vise til at noe hadde blitt gjort mellom hver iterasjon så ved bruk av en ukes iterasjon så ville det ha blitt lite å oppdatere på statusmøtene, derfor ble det bestemt at gruppen skulle bruke to ukers iterasjoner. Denne iterasjonen ville gi nok tid til å produsere produkt. Ved endt iterasjon så skulle det ha blitt produsert noe form for produkt som skulle ivareta fremgang og kvalitet i prosjektet. Det ble også mulig for produkteier og komme med noen nyttige tilbakemeldinger.

4.3 Iterasjonsplanlegging og estimering av tid

Før hver iterasjon hadde gruppen bestemt seg for å sette et hovedmål for perioden. Disse målene kunne sees på som delmål og skulle bidra til motivasjon, samt kontinuerlig arbeid frem mot det endelige produktet.

Backloggen startet med noen oppgaver som var kartlagt på forhånd. Disse oppgavene ble bestemt ut ifra oppgavebeskrivelsen og et use case som beskriver hva en analytiker skal få ut av systemet. Ettersom prosjektet var såpass flytende så var ikke alle oppgaver forhåndsbestemt og noen ble til etter hvert som gruppen så behovet for dem. Oppgaver i backlog ble lagt over i «ready»-kolonnen når disse var klare til å begynnes med. Dette gjorde at gruppen visste hva som lå klart til å begynnes på når en annen oppgave var ferdig.

Gruppens prosjektmetodikk hadde ikke strenge regler til estimering av tid utover at påbegynte oppgaver skulle være ferdig innen slutten av iterasjonen. Det ble derfor ikke satt noen krav for når de forskjellige oppgavene skulle være ferdige utover dette. Det gruppen gjorde isteden var å jobbe mot målet som var satt for hver iterasjon.

4.4 Iterasjonsgjennomgang

Etter hver iterasjon dedikeres det ukentlige møtet med produkteier til en review. På disse møtene tok gruppen opp hva som hadde blitt gjort i løpet av iterasjonen, målet de hadde og om målet ble nådd. Det ble også tatt opp utfordringer som ble støtt på underveis og hva de kunne skyldes. Avslutningsvis fikk produkteier gitt en tilbakemelding.

4.5 Rollefordeling

Ettersom gruppen har valgt scrumban som ikke har noen faste roller integrert så er ikke noen faste roller satt av prosjektmetoden. I stedet ble det naturlig at gruppen delte inn roller etter ferdigheter og kompetanse ettersom hver enkelt har forskjellige egenskaper. På bakgrunn av dette ble prosjektets oppgaver fordelt og tilpasset hvert enkelt gruppemedlem for å sikre effektivitet og kontinuerlig arbeid.

4.5.1 Produkteier

I dette prosjektet så har det vært Cato Robstad og Frank Kirkeng fra Netsecurity sin avdeling i Kristiansand som har vært produkteier. De har bistått i prosjektet og hatt ansvaret for generell oppfølging samt iterasjonsmøter.
Gruppen har også hatt en del av hovedansvaret for dette prosjektet og fikk tidlig beskjed om at mye av prosjektet var opp til gruppen å velge fremgangsmåte.

4.5.2 Ansvarlig for bedriftskontakt og teknisk

Det ble sett på som naturlig å ha kun et gruppemedlem med ansvar for kommunikasjon mellom gruppen og bedriften. Ansvaret for kontakt og kommunikasjon ut mot bedriften var Kjartan Erland. Dette ansvaret innebar å fremme gruppens utfordringer og generelle ting til bedriftens kontaktperson Cato Robstad. Da Kjartan også jobber i Netsecurity, så ble det mest praktisk å ha han som kontaktperson.

I tillegg til dette var også Kjartan teknisk ansvarlig. Denne rollen innebar å ha oversikt over de tekniske aspektene med prosjektet, og ta initiativ og ansvar til å ta kontakt ved eventuelle utfordringer for å løse dette.

4.5.3 Ansvarlig for kontakt med UiA og rapport

Dette prosjektet har foregått i samarbeid med både Netsecurity og UiA, noe som har ført til en del samtaler mellom partene. Hans Christian tok ansvar for å ha kontakt med ulike personer ved UiA, og hadde i tillegg hovedansvar for rapportens utforming og korrekturlesning.

4.5.4 Ansvarlig for tester og kvalitetssikring

Prosjektet til gruppen var ekstra avhengig av testing og kvalitetssikring grunnet mangelen på versjonskontroll. Det ble derfor naturlig å utnevne en som skulle ha ansvaret for å finne testmetoder som igjen skulle sikre kvaliteten til produktet. Ettersom Elina skal begynne å jobbe som testkonsulent, var det naturlig at hun tok på seg denne rollen.
Hennes oppgaver bestod som nevnt av å finne testmetoder, men også koordinere når og hvordan disse testene skulle utføres. Når det kom til kvalitetssikring hadde hun fokus på at gruppen overholdt de retningslinjene som var satt i forhold til prosjektgjennomføringen og at det ble opprettholdt kontinuerlig arbeid med oppgaver relatert til prosjektet.

4.6 Risikoregister

Et prosjekt er avhengig av å ha en så god oversikt som overhodet mulig over mulige risikoer som kan oppstå underveis i prosjektet. Risikoer som kan knyttes til personlige og tekniske aspekter. Da gruppen satt i gang med risikoanalysen ble det funnet en mal på nettsiden wordtemplatesonline.net som ble tatt i bruk. Denne malen ga gruppen et godt utgangspunkt for å evaluere risikoer som kunne oppstå og hvordan det ville påvirke prosjektet samt hva som skulle til for å holde prosjektet i gang hvis problemer skulle oppstå.
På Digitaliseringsdirektoratet sin nettside beskriver alt en risikovurdering burde inneholde. Hovedpunktene er som følger: identifisere, analysere og evaluere risiko (Digdir, u.åb).

Identifisere risiko

Først og fremst så vil man identifisere eventuelle risikoer og liste opp alle disse. Gruppen tok opp alle de risikoene som var mulig å komme på. Risikoene ble delt i to kategorier, personlige/menneskelige og tekniske risikoer.

Analysere risiko

Etter å ha identifisert risikoene så vil man utføre en analyse av disse. I en risikoanalyse vurderes mulige konsekvenser og sannsynligheten for at en risiko blir realitet. I risikoanalysen har gruppen valgt å benytte seg av en matrise som går fra 1 - 4 (Stakeholdermap.com, u.å).

Skalaen for sannsynlighet er dermed som følger: Usannsynlig = 1, lite sannsynlig = 2, mulig = 3 og sannsynlig = 4. For konsekvens ser skalaen noenlunde lik ut: Ubetydelig = 1, moderat = 2, alvorlig = 3 og kritisk = 4.

Evaluere risiko

Det siste som blir gjort i en risikovurdering er evaluering. Etter hver endt iterasjon gikk gruppen igjennom risikovurderingen som ble laget i starten av prosjektet. Ettersom tiden går, har risikobildet muligheten til å endres og det er dermed viktig og justere seg deretter. Det ble da gjort en ny vurdering av risikoenes sannsynlighet og konsekvens. Om det ble endringer eller gruppen hadde problemer så ble disse adressert på styringsgruppemøtene hvor både oppdragsgiver og veileder var til stede.

Som nevnt har gruppen valgt å ta i bruk en matrise som går fra 1- 4, matrisen viser hvor "alvorlig" en risiko er ut fra totalsummen risikoen blir tildelt. Risikoregisteret er en tabell hvor aktuelle risikoer har blitt ført inn og vurdert.
Dette registeret gir et illustrert risikobilde og ga gruppen oversikt over hva som hadde endret seg siden oppstarten av prosjektet. Tabell 1 på neste side er et utdrag fra gruppens risikoregister som er lagt ved i vedlegg 3.

Risiko som må taes til etterretning	Begrunnelse	S	K	T
Miste tilgang til filer og kontoer	Det å miste tilgang til filer og kontoer vil kunne føre til stans i arbeidet. Prosjektet avhenger av nettopp dette.	1	3	3
Miste arbeid pga ingen versjonskontroll	Dette kan være svært kritisk da alt vårt arbeid kan gå tapt. Ved mangel på versjonskontroll mister man muligheten til å mellomlagre arbeid og kontroll sikre dataen.	2	3	6
Dobbelt arbeid pga bruk av flere ulike maskiner	På bakgrunn av manglende versjonskontroll så vil man måtte jobbe "lokalt". Gruppen veksler mellom å jobbe digitalt på egne laptoper og dra på kontoret til bedriften og jobbe fra pc-ene der. Ettersom gruppen ikke har tilgang til det samme på sine laptoper som på bedriftens PC-er, risikerer man fra tid til annen å jobbe dobbelt opp.	3	2	6

Tabell 1: Utdrag fra risikoregisteret.

5. Kvalitet og kvalitetssikring

Kvalitet og kvalitetssikring er kanskje noe av det viktigste i et prosjekt. Disse to begrepene kan være vanskelig å definere, ut ifra hvilken type prosjekt man jobber med og hva man legger i ordet kvalitet. Spørsmålene blir naturligvis; hva er kvalitet, og hvordan måler man det? Her er det mange veier til mål, men i utgangspunktet handler det om å stille seg flere spørsmål om prosjektet, uavhengig om hva prosjektet er og innebærer.

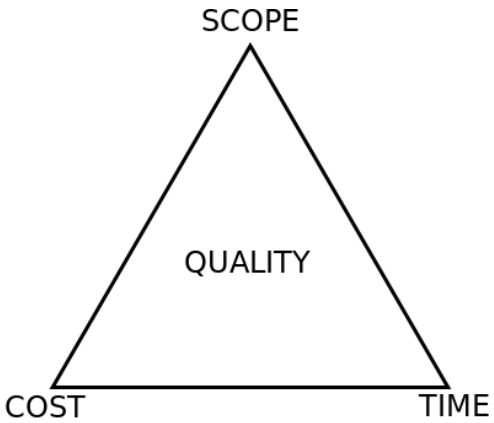
> *Hva skal løsningen være, og hvor godt gjør den det?*
> *Oppstår det feil? Er det vanskelig å bruke?*
> *Hvor rask er den, og er den sikker? Kan den endres og utvides?*

Slike spørsmål er med på å danne forståelsen av hva kvalitet er og hvordan man oppnår det. I tillegg har gruppen underliggende spørsmål som spesifiserer hvordan man skaper kvalitet. Dette er punkter som knytter kvalitet til prosjektets løsning fra start til slutt. Hvordan bygger man kvalitet inn fra starten for eksempel? Hvordan håndterer man feil i en løsning, gruppe, organisasjon, og så videre?

Hvert av disse punktene er med på å bidra til kvalitetssikring, hvor selve kvalitetssikringen kommer fra å bruke hjelpemidler til å skape kvalitet. Her er det fire punkter som er verdt å ta med seg; kvalitet i gruppen og hvordan dette gjøres både i bedriften og gruppen, prosjektstyringsrammeverk og kodestandard. Disse fire punktene er de overordnede søylene i prosjektet, som skal være med på å sikre kvalitet, og drøftes i mer detalj videre.

5.1 Kvalitet innad i gruppen

For å i det hele tatt kunne oppnå kvalitet og kvalitetssikring, så er det viktig at gruppen er på rett sted. Som allerede nevnt i kapittel 4, er Scrumban prosjektrammeverket gruppen bruker. Grunnmuren til gruppearbeidet var å danne retningslinjer og rutiner. Dette innebærer daglige og ukentlige møter. I hvert av de daglige møtene var fokuset planlegging og oppfølging. Her var målet å danne gode og effektive rutiner tidlig. Slike rutiner innebar f.eks. delegering av arbeidsoppgaver, lage mal på dokumenter, innleveringer osv.



Figur 2: The project management triangle, 2007, av Mapto.

(https://commons.wikimedia.org/wiki/File:Project-triangle-en.svg). Offentlig eiendom.

Kvalitet handler også mye om prioriteringer, noe som illustreres godt av figuren over. Trekanten skal illustrere forholdet mellom de tre viktigste begrensningene som påvirker kvalitet i et prosjekt, og sier at kvaliteten på et gjennomført arbeid er begrenset av prosjektets kostnad, tidsfrister og omfang. Det er i utgangspunktet prosjektleder sitt ansvar å finne det mest optimale forholdet mellom disse tre begrensningene, hvor endring i forholdet mellom disse må tas hensyn til for at dette ikke skal gå ut over kvaliteten på prosjektet.

5.2 Kvalitetssikring i samarbeid med bedrift

Ettersom prosjektets mål var å oppfylle kravene stilt i prosjektbeskrivelsen, var det kritisk å komme i samtale med bedriften for å få en bedre forståelse av hvilke utfordringer man står overfor, og hvordan dette kan løses best mulig. I dette prosjektet har gruppen vært heldig med at det hele foregår i fagfeltet og hverdagen til bedriften, og de har dermed god teknisk forståelse og et annet perspektiv enn en mindre IT-kyndig bedrift ville hatt. Dette gjør det mulig å diskutere konkrete løsninger på et lavere nivå, og gjør det enklere å opprettholde klar og tydelig kommunikasjon for at man kan ha en felles forståelse om forventninger mellom gruppen og bedriften.

For å oppnå og opprettholde god kommunikasjon med oppdragsgiver har rutiner i form av ukentlige møter vært verdifulle, da det gir gruppen mulighet til å reflektere over ukens arbeid, få veiledning og annen hjelp fra oppdragsgiver. Møtene er og med på å tydelig kommunisere hva som er blitt gjort, hvordan det er gjort, og videre planlegging i plenum. Møtene er viktig for at oppdragsgiver skal kunne komme med innspill om hva som er blitt gjort eller burde gjøres, og for å passe på at man er på samme bølgelengde når det kommer til forventninger om omfang og kvalitet.

Gruppen mener at god kommunikasjon med bedriften har vært det viktigste tiltaket for kvalitetssikring gjennom hele prosjektløpet. Et konkret eksempel på dette er da gruppen skulle sette opp automatisk tildeling av alvorlighetsgrad knyttet til hver incident. Dette kan gjennomføres på ulike måter i kode, og fikk gode innspill gjennom god kommunikasjon med bedriften. Fremgangsmåten som ble tatt i bruk var basert på tidligere erfaring hos en av bedriftens sikkerhetsanalytikere, som visste dette ville dekke behov på en god måte. Det var en enkel implementasjon som tilordnet verdier for alvorlighetsgrad fra Rubrik til XSOAR sitt format.

5.3 Scrumban

Scrumban er en hybrid mellom scrum og kanban, og baserer seg på to enkle konsepter. Hvert gruppemedlem valgte en oppgave de ønsket å arbeide med, og er nødt til å gjøre seg ferdig med oppgavene de har påtatt seg før man kan ta på seg flere eller gå videre. Dette hjelper med å forhindre at man blir overveldet av for mange pågående oppgaver på samme tid, og gjør det enklere å holde fokus og være målrettet. For å delegere oppgaver har gruppen brukt Microsoft To Do som er en slags digital tavle for å planlegge, velge og gjennomføre oppgaver. Estimering av tid på oppgavene fulgte iterasjonene, der målet var å gjennomføre oppgavene satt per iterasjon, men da Scrumban er en agil metode var det rom for fleksible estimeringer.

Scrumban viste seg å være veldig effektivt for gruppen som et verktøy som bidro til godt kontinuerlig arbeid. Ifølge en forskningsartikkel fra Universitet i Bucuresti, så hevder de også at Scrumban som prosjektmetodikk kan optimalisere gruppens innsats mot kvalitetssikring. Da Scrumban sin sterke side er agilitet hadde gruppen mulighet til å jobbe mer fleksibelt med oppgavene (Stoica, et al., 2016, s. 11). Ettersom Netsecurity ikke har noen etablerte prosjektstyringsmetoder, passet Scrumban godt inn for gruppen og bedriften ettersom det var klare retningslinjer å følge. Det at bedriften også kunne følge med iterasjonenes intervaller ga mulighet for den hyppige kontakten og kvalitetssikringen gruppen var ute etter.

5.4 Kodestandard

I dette prosjektet går mye av gjennomføringen ut på å skrive på integrasjonen og lage playbooks i XSOAR, som benytter seg av programmeringsspråket Python. I tillegg til dette har XSOAR en egen standard for oppsett på hvor og hvordan kode blir skrevet. Utfordringen for gruppen var dermed todelt, ettersom ingen av oss hadde noen særlig tidligere erfaring med Python, i tillegg så man seg nødt til å sette seg inn i XSOAR sitt rammeverk, som i hovedsak baserte seg på to ting, integrasjonen og playbooken.

En integrasjon består av to hoveddeler, selve koden, og en rekke ulike innstillinger, som API-innstillinger, parametere, kommandonavn og output. Parametere, kommandonavn og output må defineres og gis funksjon i Python, men man må også fylle ut informasjon om koden for at XSOAR skal kunne benytte seg av den på korrekt vis. Og ettersom playbook er en måte å automatisere funksjoner på i incidents, så er det ekstra viktig at oppsettet til playbooks er så standardisert som mulig. Det skal med andre ord være fleksibelt nok til å bruke playbooks i flere bruksområder, basert på incident-type for eksempel.

Dette er et av aspektene med kvalitetssikring som gruppen har sammen med bedriften, at tjenesteelementet skal kunne videreutvikles og skalerer videre.
Selv om det gjøres endringer i skript og integrasjon, skal playbook fremdeles gjøre jobben med å mappe data til felter i incidents.

6. Prosjektgjennomføring

I dette kapittelet blir gjennomføringen av prosjektet presentert. Arbeidet gruppen har utført vil bli drøftet i sin helhet, samt hvilke utfordringer som har oppstått underveis og hvordan disse har blitt mestret. I hele prosjektperioden har gruppen benyttet seg av agil prosjektmetodikk som strukturen på rapporten reflekterer. Hver iterasjon vil også bli gått igjennom.

6.1 Analyse

I en analyse skal man knytte sammen tilegnet informasjon fra et område. Denne informasjonen kommer fra innsamlet materiale og undersøkelser(kilde). Gruppen har i dette prosjektet gjort bevisste valg for å sikre at forventningene til produktet blir som forventet. For å sikre dette så var det viktig å innhente informasjon fra alle som kunne være aktuelle brukere av produktet. Informasjonsinnhentingen besto i stor grad av uformelle intervjuer av aktuelle brukere.

6.1.1 Målgruppe

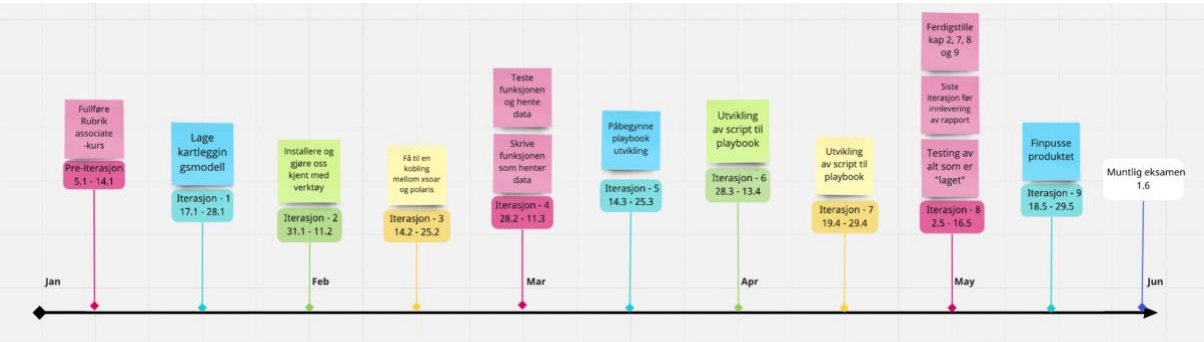
Det å kartlegge hvem produktet er ment for er viktig for å sikre at det blir som forventet. I dette prosjektet er en målgruppe dem som skal ta i bruk produktet. Gruppens målgruppe har helt fra begynnelsen av prosjektet vært bedriftens sikkerhetsanalytikere, ettersom det er dem som skal ta i bruk det nye tjenesteelementet som blir implementert i deres orkestreringsverktøy. Produktet er et nytt tjenesteelement som bedriften skal ta i bruk for å levere et bredere tjenestetilbud til sine kunder.

6.2 Design

For å kartlegge designet måtte gruppen dele oppgaver(tasks) i mindre deler basert på hvor oppgaven tilhører. I første utkast kartlegges high level design, som vil si alle brikkene i det gruppen ser for oss tjenesten avhenger av. Én brikke er kundedata fra Office/Microsoft365, Exchange, Teams osv. Denne dataen sendes til Rubrik for backup. Dataen som er i backup vil XSOAR kunne hente metadata fra for å se at alt fungerer som det skal. Dette er essensen i prosjektet, at analytikere på Netsecurity skal kunne se på XSOAR at alt fungerer i backup på Rubrik.

6.3 Iterasjoner og implementeringer

I dette prosjektet har gruppen benyttet seg av scrumban-rammeverket med to ukers varighet på iterasjonene. Hver iterasjon har hatt et hovedmål som gruppen har fokusert på, disse skal bli presentert med en refleksjon over hva som har gått bra og hva som kunne ha blitt gjort bedre. Gruppen laget en tidslinje i verktøyet Miro som viste hele perioden med iterasjonene og deres mål. Skjermbilde 6 viser en overordnet oversikt over alle iterasjonene og gir en illustrasjon på tidsaspektet til prosjektet.



Skjermbilde 6: Oversikt over iterasjonene og målene for disse.

6.3.1 Pre-iterasjon - Planleggingen (5. januar - 14. januar)

Før gruppen gikk i gang med iterasjon 1 så ble det bestemt at man skulle å gjennomføre en pre-iterasjon. Pre-iterasjonen skulle bidra til at gruppen fikk en oversikt over prosjektomfanget og deretter planlegge de iterasjonene som gruppen skal gjennom.
En slik pre-iterasjon bidro til at alle i gruppen ble enig om hvordan prosjektet skulle gjennomføres fra start til slutt.

Det første gruppe tok for seg var planleggingen av arbeidsmengde og prosjektstyringen. I denne perioden ble også mål for de kommende iterasjonene satt.
Videre ble det bestemt hvilken prosjektmetodikk som skulle følges. Gruppen tok også et kurs knyttet til Rubrik og startet med strukturingen av rapporten.

6.3.2 Iterasjon 1 - Kartlegging av produktet (17. januar - 28. januar)

I iterasjon 1 gikk det mye tid til å kartlegge flyten til produktet. Bakgrunnen for dette var at gruppen skulle forstå hvordan produktet skulle fungere for så å kunne planlegge og strukturere oppgavene til backloggen ut ifra dette. Figuren som ble vist frem i kapittel 2 (se figur 1) ble vist frem til produkteier som ga en tilbakemelding på at gruppen hadde tenkt riktig om hvordan flyten skulle være. I tillegg til denne så laget gruppen en use case (se tabell 2 under) for å kunne se for seg hvordan produktet kunne anvendes samt tenke ut oppgaver til backloggen.

Da gruppen begynte for alvor med oppgaveoversikten så kategoriserte gruppen noen kolonner for å måle fremgang. Som nevnt under kapittel 4.3 så startet backloggen med noen oppgaver som var kartlagt på forhånd. Underveis i prosjektet ble flere oppgave lagt til og disse ble laget på bakgrunn av behov og målet for den iterasjonen gruppen var i på det tidspunktet. Oppgaver som var klare til å begynnes på tidlig ble flyttet fra backloggen og over i kolonnen som het “ready”. Hvert gruppemedlem tok en oppgave fra “ready”-kolonnen og plasserte den i “In progress” når de hadde startet på den oppgaven de hadde valgt.

Navn på case	Hendelses håndtering av sikkerhetsanalytiker
Beskrivelse	Analytiker ønsker å innhente god nok informasjon for å oppklare hendelser
Brukertype	Analytiker
Organisasjonsfordeler	Sikker (og rask) hendelses håndtering for kunder = fornøyde kunder
Antagelser	For at analytiker skal kunne utføre jobben så må Rubrik sitt system fungere.
Hyppighet av bruk	Brukes hver dag.
Utløsende faktorer	Noe galt skjer i backup eller filsystem.
Forutsetninger	Systemet må ha en sikkerhetskopi som kan brukes.
Utfall	Filsystemet eller individuelle filer blir gjenopprettet.
Fremgangsmåte	Steg 1. Analytiker ser hendelse av type X Steg 2. Analytiker innhenter relevant informasjon på hendelsen Steg 3. Analytiker tar en avgjørelse på saken og følger opp selskapets prosedyre basert på utfall av hendelsens type.
Alternativ fremgangsmåte	Vedlikehold av system. Direkte innlogging på backup tjenesten.

Tabell 2: Use case

I denne iterasjonen knyttet gruppen lærdom til viktigheten av nøye kartlegging og planlegging av et prosjekt. Det viktigste gruppen tok fra denne lærdommen, var å ha regelmessig kontakt med prosjekteier og kontaktperson fra bedriften gjennom kartleggingsprosessen. Dette førte til en form for kvalitetssikring også, da alle parter i prosjektet tok del i prosessen underveis. Mot slutten av iterasjonen, så lagde gruppen en risikomatrise som skulle kartlegge mulige risikoer som kunne oppstå underveis i prosjektet. Dette bestemte gruppen seg for å lage på bakgrunn av samtaler med tidligere bachelorstudenter som ga uttrykk for at det kunne være lurt.

6.3.3 Iterasjon 2 - Installere og bli kjent med verktøy (31. januar - 11. februar)

Målet for denne iterasjonen var å få alle nødvendige verktøy og tilganger på plass før det planlagte arbeidet med integrasjonen. Det ble også gjennomført et møte med en Rubrik representant for å høre om hva slags tanker de kunne ha om prosjektet og mulig funksjonalitet.

Det viste seg at det å få installert alt av programmer og få tilgang til systemer ikke skulle være så lett for gruppen. Underveis i iterasjonen måtte gruppen begynne å tenke alternativt for å få gjort noe nyttig knyttet til prosjektet. Gruppen begynte derfor tidligere å lese seg opp på det som fantes av dokumentasjon på GraphQL som er det API-et gruppen har brukt til å koble sammen Rubrik Polaris og XSOAR-plattformen. Å lese seg opp på API var noe som først skulle startes på i iterasjon 3 som var den første iterasjonen planlagt til integrasjonsarbeid. Gruppen begynte også smått å dokumentere i rapporten om prosjektmetodikken scrumban som de hadde valgt.

I denne iterasjonen lærte gruppen at det som er planlagt ikke alltid går etter planen og det er derfor viktig å være forberedt på og jobbe med alternative oppgaver så prosjektet ikke ble satt på vent. Fokuset for neste iterasjon ble på grunn av uforutsette årsaker å få på plass det gruppen trengte av tilganger og programmer samt starte på arbeidet med integrasjonen.

6.3.4 Iterasjon 3 - Integrasjonsarbeid (14. februar - 25. februar)

I Iterasjon 3 var målene våre å arbeide videre med integrasjonen, samt få tilgang til vCenter å kunne ta i bruk VM-er og backups. Det ble i tillegg brukt en del tid på å bli kjent med Rubrik Playground, som er et grensesnitt for å utforske og eksperimentere med GraphQL og det som var tilgjengelig av dokumentasjon.

I den andre uken av iterasjonen deltok gruppen på Camp Rubrik, som var en workshop for å lære seg å bruke produktet. Gruppen begynte også å eksperimentere med å implementere to ting i XSOAR, en «incident type» og en «fetch-funksjon». Dette skjedde parallelt med arbeid hvor man skrev dokumentasjon for den endelige rapporten.

Gruppen har i løpet av denne iterasjonen sett hvor viktig det er med god dokumentasjon, da de selv har hatt problemer med dette. Her tok gruppen med seg at det er både nyttig og lurt å dokumentere det arbeidet de gjør nøye, det de utvikler skal tross alt bli tatt i bruk av andre som ikke har samme bakgrunnskunnskap som dem.

6.3.5 Iterasjon 4 - Integrasjonsarbeid (28. februar - 11. mars)

I iterasjon 4 fortsatte arbeidet med integrasjonen, med fokus på funksjonen å hente data fra Polaris. Her var målet å skrive en “fetch-funksjon”, til å hente data fra Polaris som gruppen hadde generert som test-data. Metoden for å generere test-data var utfordrende, da dataen i utgangspunktet skulle være av typen feilmeldinger; “Error i Backup” som et eksempel. Gruppen bestemte seg heller for å lage falsk-positive data, dvs. “Fullførte Backup”, da dette var en mye bedre metode for å teste funksjonen å hente data.

Siste del av iterasjon 4 var å fikse metoden XSOAR håndterte datomarkering på alarmer. For at hendelser skulle bli laget, måtte datomerking være korrekt dato på alarmen når den oppsto og hvordan XSOAR formaterte dette. Dette viste seg å være problematisk en stund, da det oppsto flere feilmeldinger og ingen av alarmene ble generert. Etter flere møter med teknisk ansvarlig fra bedriften, viste det seg å være en feil i test-miljøet på XSOAR som forårsaket dette. Dette resulterte med at gruppen måtte migrere arbeidet over til produksjonsmiljøet.

I denne iterasjonen, så tok gruppen stor lærdom fra kontaktpersonen i bedriften. Det å ikke være redd for å stille spørsmål om ting man lurer på, når man trenger hjelp, balansen mellom å prøve ting selv før man spør om hjelp osv. Gruppen benyttet metoden å bruke et gjennomsnitt på ca. 5 dager, hvor hovedfokuset var å prøve å finne ut ting selv, før man eventuelt tok kontakt for å få hjelp. Dette er for å ikke skape lange perioder hvor man ikke har fremgang på "unødvendige" ting.

6.3.6 Iterasjon 5 - Playbook (14. mars - 25. mars)

Iterasjon 5 var to uker der gruppen fikk gjennomført mye godt arbeid innen mange ulike områder. Det mest sentrale for iterasjonen var arbeidet med playbooks i XSOAR. Dette startet som kartlegging sammen med Netsecurity, og gruppen fikk tid til å planlegge og starte utvikling mot slutten av iterasjonen.

Etter mye trøbbel med tidsstempel på sikkerhetshendelser i XSOAR, og mange samtaler og feilsøking med hjelp fra Netsecurity, bestemte gruppen oss for å teste integrasjonen i produksjonsmiljøet. Dette viste seg å delvis løse problemet, og lot oss gå videre med prosjektet. Ellers gikk mye av arbeidet på dokumentasjon, rapport og annen skriving.

Denne iterasjonen ga gruppen flere gode erfaringer. Fra tidligere iterasjoner var det tydelig at grunnmuren i prosjektet var god kommunikasjon og det var derfor kritisk at gruppen opprettholdt den stabile kontakten med oppdragsgiver. Utfordringen knyttet tidsstempel i XSOAR var også en god erfaring i feilsøking av et system, og ga oss innsikt i hvordan Netsecurity arbeider.

6.3.7 Iterasjon 6 - Playbook (28. mars - 13. april)

I denne iterasjonen var det også fokus på playbook-utvikling da dette var en oppgave som var noe omfattende og krevde mer tid enn kun én iterasjon. Det som skulle sees nærmere på i denne perioden var scriptet som playbooken kjører.

Oppgaven med å mappe ut hva som skulle være med i scriptet var det som tok mest tid. Så snart dette var gjennomført, gjenstod det bare å bestemme hva som skulle vises i incident-visningen, også kalt Incident Layout. Det er her informasjonen som sikkerhetsanalytikerne ser fra selve alarmen dukker opp. Parallelt med skriptarbeid så jobbet gruppen videre på rapporten for å sikre at den kontinuerlig var oppdatert med den nyeste dokumentasjonen som var tilgjengelig.

For gruppen var det viktig å sikre kontinuitet i arbeidet i denne iterasjonen. Det var derfor viktig at alle på gruppen hadde noe å gjøre til enhver tid og var opptatte av å vedlikeholde kommunikasjonen seg imellom og med oppdragsgiver.

6.3.8 Iterasjon 7 - Playbook (19. april - 29. april)

Iterasjon 7 var den siste iterasjonen med fokus på playbook. Bakgrunnen for at playbook hadde 3 iterasjoner dedikert til seg var fordi playbook utviklingen besto av flere aspekter som tok tid. Kapittel 6.3.6 fokuserte på kartleggingen av hva som skulle med, kapittel 6.3.7 hadde fokus på scriptet som playbook skulle kjøre.
I denne iterasjonen er fokuset på å få med den siste viktige informasjonen som må med i incident-visningen samt lese den nye oppdateringen til Rubrik for å se om det er noe ny funksjonalitet som kan implementeres ved en senere anledning av Netsecurity selv.

Den informasjonen som manglet i incident-visningen var alvorlighetsgraden på hendelsen, denne var det litt jobb å få til for at den ble riktig, men det ordnet seg siste dagen i iterasjonen. Det nye oppdateringen til Rubrik gir mulighet for å vise Cluster-Location i incident-visningen, men denne informasjonen er ikke viktig for å håndtere en feilet backup og er derfor ikke lagt til i denne omgang.

Det ble også økt fokus på rapporten ved denne iterasjonen ettersom innleveringen av denne nærmet seg. Gruppen fikk dokumentert utviklingen av playbooken og videre dokumentert viktig informasjon om prosjektet.

6.3.9 Iterasjon 8 - Testing og ferdigstille rapport (2. mai - 16. mai)

I iterasjon 8 tok gruppen for seg de siste testene og skrev ferdig rapporten. Dette var dermed den siste iterasjonen før rapporten skulle leveres og prosjektet nærmet seg slutten.

Gruppen fikk gjennomført en akseptansetest som var kombinert med brukertesten. Resultatet av disse testene ga gruppen bekreftelse på at de har utviklet et tjenesteelement som kan brukes. I tillegg til testingen hadde gruppen fokus på å bruke veileder for å få viktige tilbakemeldinger som kunne heve presentasjonen av prosjektet. Veiledningen var svært nyttig og ga gruppen den lille ekstra motivasjonen til å dokumentere prosjektet på en så god måte som mulig.

Iterasjonen ble rundet av med at gruppen finpusset rapporten og fikk lagt ved uttalelsen fra prosjekteier.

6.3.10 Iterasjon 9 - Finpussing av produkt (18. mai - 29. mai)

Iterasjonen 9 er gruppens siste da dette er perioden mellom der rapporten er levert og den muntlige fremføringen. Hovedfokuset vil naturlig være å finpusse tjenesteelementet og få det så bra som mulig opp imot den muntlige eksamen.

7. Tester

I dette kapittelet vil gruppen ta for seg de ulike testmetodene som ble tatt i bruk under prosjektet samt hvilke modeller som hadde vært mulig i et slikt prosjekt. Gruppens prosjekt består av mye testing da versjonskontroll som nevnt ikke er tatt i bruk. Det er variasjon blant prosjektmetodikker på hvordan og når en kan utføre tester.

Den ene metoden er testdrevet utvikling. Testdreven utvikling har fokus på å lage enhetstest-caser før man utvikler noen form for kode (Unadkat, 2021). Om gruppen skulle ha programmert hele tjenesteelementet fra bunnen av ville denne metoden ha vært nyttig å bruke.

En annen modell som blir brukt i systemutvikling er V-modellen. Denne modellen er i motsetning til en agil modell delt i to faser. Hvorav den ene fasen dekker verifisering, mens den andre dekker validering. I denne modellen er det vanskeligere å få gjennomført testing underveis i prosessen, ettersom testingen gjøres imot slutten av utviklingen eller når det er ferdig utviklet (GeeksforGeeks, 2019). Denne metodikken minner om waterfall ettersom den også følger de samme stegene (GeeksforGeeks, 2020). Gruppen kunne ha brukt denne modellen om systemet ikke måtte testes underveis i prosjektet.

Ettersom gruppen har en agil prosjektmetodikk som innebærer endringer og testing underveis i prosjektet, så var det mest naturlig å følge et løp hvor testing ble gjennomført ved flere anledninger gjennom hele prosjektet. Dette var for å sikre og levere et fungerende produkt. Videre bestemte gruppen seg for hvilke tester som skulle gjennomføres og deretter kategorisere dem som funksjonelle og ikke-funksjonelle. Først vil de funksjonelle testene bli skrevet om før de ikke-funksjonelle blir presentert.

Gruppens mål med testene er å få testet om man får opp en sikkerhetshendelse, i XSOAR kalt incident, og at den laster inn nok informasjon til at en sikkerhetsanalytiker kan håndtere den.

7.1 Funksjonelle tester

Ved funksjonell testing bekreftes funksjonaliteten til integrasjonen som testes. Det skal sikres at funksjonene tilfredsstiller de kravene som er spesifisert. Viktigheten med funksjonell testing er å få beskrevet hvordan utførelsen og oppførselen til integrasjonen er. Funksjonelle tester utføres før de ikke-funksjonelle testene, dette for å forstå hvordan funksjonene i systemet fungerer (Softwaretestinghelp, 2022).

7.1.1 Enhetstesting

Enhetstesting er som oftest første test som gjennomføres og brukes for å sikre at individuelle komponenter i et system på kode-nivå fungerer som de er bestemt til. En slik test utføres av utviklerne selv. I dette prosjektet har gruppen brukt denne metoden hyppig. Dette var for å sikre at de essensielle funksjonene i integrasjonen fungerer, og var klare til å testes i bolker med en integrasjonstest (Aebersold, u.å).

Tidsstempler er kritiske for at XSOAR skal kunne hente inn og knytte data til hendelser, og utfordringer med ulik formatering på tidsstempler mellom XSOAR og andre tjenester førte til at testing av tidsstempler var sentralt i gruppens arbeid gjennom store deler av prosjektet. Ettersom tidsstempel er nødvendig for å generere incidents i XSOAR finnes det boilerplate-kode og dokumentasjon som gruppen tok utgangspunkt i. For å sikre at tidsstemplene fungerte som forventet benyttet gruppen seg av et debugging-verktøy i XSOAR som muliggjør å nullstille dette stempelet for å sjekke om integrasjonen hentet alarmer markert med dato.

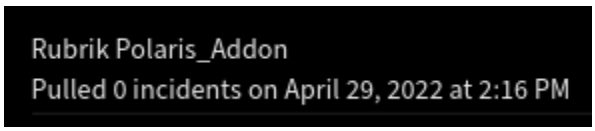
For å teste endringer på incidents, så kan man gjøre det via War Room. Dette kan gjøres ved å skrive kommandoen “![integrasjonsnavn]” for å enkelt teste endringer i incident view uten å måtte kjøre hele playbooken på nytt som genererer incidenten. Dette skriptet er der dataen fra integrasjonen kommer fra og blir formatert inn i et incident view.

Man kan gjøre endringer i skriptet og kalle det i War Room-funksjonen i XSOAR for å se endringene direkte i en incident. Ved å kalle skriptet som kjører på incidenten vil informasjonen bli formatert rett i Incident View, noe som gjør det mulig å sjekke at de ulike feltene blir formatert med riktig informasjon.

7.1.2 Integrasjonstest

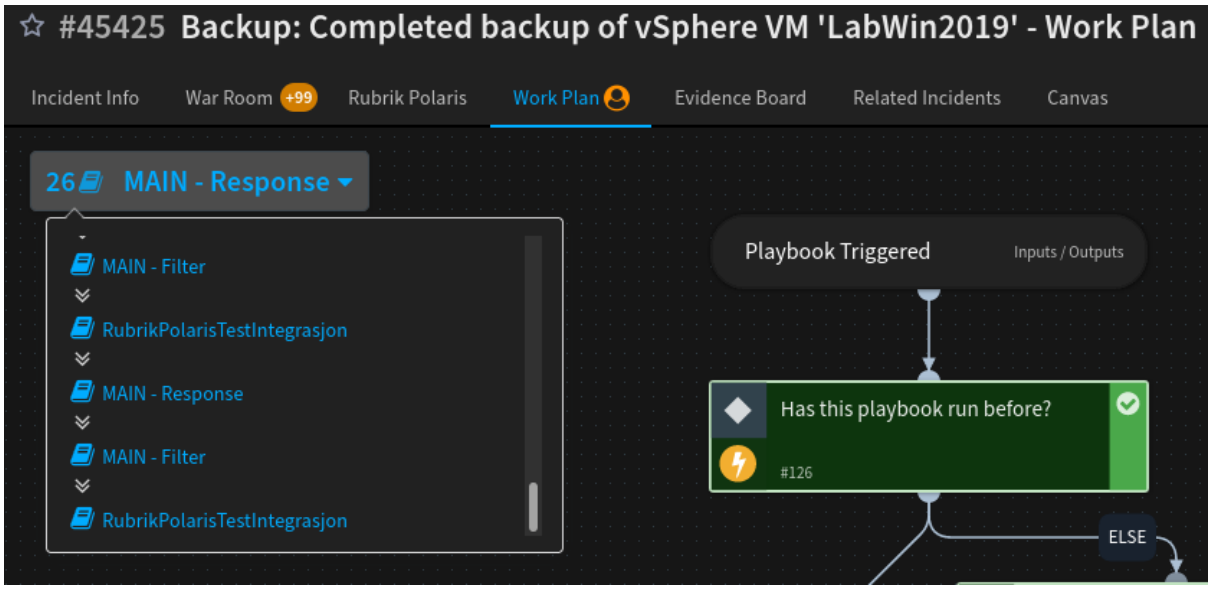
Integrasjonstesting er tester som gjennomføres etter enhetstesting fordi flere funksjoner er nødt til å testes sammen i bolker. Dette skal sikre at flere segmenter av systemet fungerer som forventet sammen. Slike integrasjonstester er ofte utformet etter brukerscenarioer. (Softwaretestinghelp, 2022).

I tjenesteelementet så finnes det to hoved-"integrasjoner" som inneholder flere funksjoner. I denne integrasjonstesten, så måtte gruppen teste at playbooken fungerer og gjør jobben riktig, samt koden i XSOAR integrasjonen. Først ut var integrasjonen, da dette er den koden som faktisk henter Rubriks backupdata og setter timestamp på når hendelsen har skjedd. Som man kan se i skjermbildet under, kan man lett teste at dette funker ved at XSOAR vil gi beskjed dersom funksjonen i integrasjonen fungerer. Her kan man for eksempel gjøre endringer på spørringen i integrasjonen som henter data for å se hvordan XSOAR reagerer, som i skjermbilde 7, som viser at det ikke finnes en feil i noen backups.



Skjermbilde 7: Viser hvor mange incidents som er hentet.

Neste del av integrasjonstesten var playbook. Dette ble gjort ved å kjøre playbooken opp mot integrasjonen for å se om dataen som blir hentet faktisk blir formatert til riktig felt in incidenten. I skjermbildet 8 på neste side kan man se at man har kjørt playbooken et par ganger for å sjekke at den mapper data til incidenten.



Skjermbilde 8: Viser at playbook er kjørt for å sjekke om den mapper data til incident.

7.1.3 Akseptansetest

Den siste funksjonelle metoden for testing er akseptansetesting og blir brukt til å vurdere om integrasjonen er klar for levering. Denne testen skal sikre at integrasjonen møter alle satte kriterier og sluttbrukers behov. En slik test krever at integrasjonen testes både internt og eksternt. Med internt menes gruppen selv og eksternt er sikkerhetsanalytikerne som skal ta i bruk systemet. Ved å teste sammen med sluttbruker får gruppen verdifulle tilbakemeldinger som kan bedre eventuelle mangler (Aebersold, u.å).

I forkant av testen ba gruppen kontaktpersonen i bedriften om å spesifisere noen akseptansekrav som akseptansetesten skulle ta utgangspunkt i. Disse ga grunnlaget for hva han skulle forvente og få ut av tjenesteelementet.

Akseptansekravene er som følger:

- Integrasjonen kobles til eksternt system (Rubrik Polaris) på API fra Netsecuritys SOAR-plattform (Security Orchestration, Automation and Response).
- Integrasjonen skal hente ut alarmer fra kilden (Rubrik Polaris) til Netsecuritys SOAR plattform som brukes for hendelseshåndtering av drifts- og sikkerhetshendelser.
- Integrasjonen skal populere felt i SOAR for å gi analytikerne som håndterer alarmene riktig og relevant informasjon om hendelsen. Med relevant informasjon menes blant annet:
    - Tid for hendelse
    - Hva som har skjedd
    - Kritikalitet av hendelse
- Integrasjonen skal følge oppsettet og utforming til tidligere integrasjoner, derav Playbooks, Automasjoner/scripts, Layout av alarmene.

Resultatene gruppen fikk ved akseptansetesten var ikke overraskende. Alle forhåndsatte krav blir dekket og hele tjenesteelementet fungerer som forventet. Akseptansetesten ble gjennomført ved at kontaktperson gikk gjennom hele tjenesteelementet mens gruppen hørte på hva han ga av tilbakemeldinger og om kravene ble oppfylt. Også kontaktperson mener at alle krav er møtt.

7.2 Ikke-funksjonelle tester

Ikke-funksjonelle tester skal avdekke aspekter ved systemet som ikke blir sett på ved funksjonell testing. Dette kan være brukervennlighet, ytelse av systemet eller pålitelighet. Bakgrunnen for de ikke-funksjonelle testene er å få sjekket om bruker er fornøyd ved bruk av systemet (Softwaretestinghelp, 2022).

7.2.1 Kompatibilitetstesting

Kompatibilitetstesting gjennomføres for å måle hvordan integrasjonen fungerer i de miljøene den er ment til å brukes. Slik testing skal sikre at systemet funker som forventet hos sluttbruker. Da integrasjonen kun skal brukes til å hente informasjon mellom to satte systemer vil testen kun kvalitetssikre at kompatibiliteten er slik den skal være mellom disse (Aebersold, u.å).

Gruppen gjennomførte denne testen noen dager før akseptanse- og brukertesten ble gjort. Det å sjekke kompatibilitet før test hos sluttbruker er viktig for å sikre at akseptanse- og brukertest blir gjennomført på en god måte og at sluttbruker får testet hele systemet. Resultatet av viste at tjenesten er kompatibel med de systemene den skal brukes sammen med, noe som var forventet da tjenesteelementet har gått igjennom både enhetstester og integrasjonstester underveis i prosjektet.

7.2.2 Brukertesting

Den første ikke-funksjonelle testen som gruppen tok i bruk på sitt system var brukertesting. Brukertesting er den testen som sluttbruker utfører for å teste om designet av systemet møter arbeidsflyten som forventet. Da denne testen er ofte utført i kombinasjon med akseptansetester tok gruppen på bakgrunn av dette og gjennomførte disse to testene sammen (Aebersold, u.å).

Kontaktperson skulle gjennomføre akseptansetesten og fungerer også som bruker, derfor ble disse testen kombinert. Forskjellen mellom testene er hva kontaktpersonen skal fokusere på. Som nevnt under akseptansetest kapittelet så fokuserer den testen på funksjonalitet i tjenesteelementet, mens under brukertesten er fokuset på det mer visuelle og om det gir flyt i arbeidet. Da kontaktperson allerede er vant med plattformen som tjenesteelementet er integrert i så vil fokuset primært være på det visuelle inne på incident visningen.

Kontaktperson synes incident-visningen ser bra ut og slik det burde være. Flyten i arbeidet med tanke på det visuelle er bra og det skal fungere fint å bruke dette tjenesteelementet.

7.3 Resultater

Etter å ha gjennomført alle tester viste det seg at gruppens vurdering av tjenesteelementet møtte kravene til prosjektbeskrivelsen, og var lik kontaktpersonens vurderinger. Resultatet av alle testene har ført til en godkjent akseptansetest, hvor hensikten var å få tjenesteelementet til en god nok standard for å bruke mot kunder. Her ble det nevnt av kontaktpersonen at oppgaven dette tjenesteelementet skulle utføre var bra, siden målet var å få driftskunder over til Rubrik for å automatisere vedlikeholdsarbeidet på driftssiden. I stedet for å rutinemessig sjekke status på servere, kan sikkerhetsanalytikerne automatisere dette gjennom XSOAR. Funksjonene som ble testet i de funksjonelle testene, viste seg da å holde mål med kravene.

De ikke-funksjonelle testene derimot var litt annerledes, da funksjon er viktigere enn direkte brukervennlighet. Disse testene gikk rett og slett ut på å sjekke at relevant informasjon ble presentert skikkelig på oversikten i en incident. Som nevnt, så er tjenesteelementet fokusert på driftssiden, og den funksjonen elementet skal gjøre, er å gi beskjed om noe går galt. Via testene ble kravene møtt av kontaktpersonen. Gruppen sjekket også at kravene tilfredsstilte use case, som kort oppsummert er som følgende: incident dukker opp i XSOAR og incidenten viser nok informasjon for at sikkerhetsanalytiker kan ta en avgjørelse videre.

8. Refleksjon

I dette kapittelet vil gruppen ta for seg en refleksjon av prosjektet. Først vil det reflekteres over prosjektmetodikken Scrumban deretter vil gruppen reflekter rundt videre utvikling av prosjektet. Det vil også bli gitt en refleksjon på utfordringene som oppsto i perioden og avslutningsvis vil det også reflekteres over erfaringer og lærdommer gruppen har hatt.

8.1 Scrumban

I dette prosjektet har rammeverket Scrumban vært svært viktig for å sikre god prosjektstyring. Det har vært med på å sikre kvalitet, fremgang og god oversikt i prosjektperioden. Ved å ha mål for hver iterasjon samt sette mål for hver arbeidsdag så har alle i gruppen hatt oversikt over hva som skal jobbes mot til enhver tid. Ved hver endt arbeidsdag hadde gruppen et kort møte for å planlegge neste dag. Dette ga gruppen kontroll over hva som måtte begynnes med av diverse oppgaver samt at det sikret kontinuerlig arbeid. Bakgrunnen for at gruppen hadde et slikt regime var for å holde prosjektet oversiktlig og strukturert. Iterasjons mål og planlegging har vært viktig for at iterasjonene skulle være gjennomførbare, slik at når det ukentlige møtet med kontaktperson i bedrift kom så hadde gruppen erfaringer og lærdommer å komme med. Det å ha praktisert Scrumban i bachelorprosjektet har gitt gruppen en god forståelse for metodikken og den bruk samt hvor greit det er å jobbe agilt i et prosjekt hvor ikke alt er spesifisert fra start.

8.2 Videre utvikling

Da tjenesteelementet var ferdig utviklet og godkjent av produkteier, satt gruppen igjen med en tjeneste produkteier skal ta i bruk. Som nevnt tidligere i denne rapporten så skulle gruppen ta å utvikle denne tjenesten med utgangspunkt i en ny oppdatering fra Rubrik. Denne var forventet i februar, men kom først i april. Gruppen har da måtte tatt utgangspunkt i den informasjonen som allerede var ute fra Rubrik når de utviklet tjenesteelementet.
Når den nye oppdateringen kom, leste gruppen over denne for å kunne se om de fant noe som ved en senere anledning kan implementeres i tjenesteelementet. Jobben videre var da å dokumentere overfor prosjekteier hva slags funksjoner og informasjon som var nyttig å ta med videre. Slike funksjoner og data var uansett utenfor scopet til prosjektet, men hensikten var å danne et bilde for videreutvikling for bedriften.

8.3 Utfordringer

Gjennom prosjektet har gruppen møtt på forskjellige utfordringer som har variert i betydning og omfang for gruppen. I dette delkapittelet vil gruppen ta for seg de ulike utfordringene som oppsto i løpet av perioden og hvordan disse ble håndtert.

8.3.1 Tilgang på systemer

Det å få tilgang på diverse systemer har vært en utfordring fra starten av prosjektet. Konsekvensene av dette førte blant annet til at gruppen var nødt til å arbeide med andre deler av prosjektet i noen tilfeller fordi gruppen ikke hadde fått korrekt tilgang. Gruppen var da svært opptatt av å gjøre noe annet som kunne knyttes til prosjektet og som fortsatt ville gi kontinuerlig fremgang. Dette kunne være å skrive rapport eller lete etter god dokumentasjon av systemene som skulle brukes. På tross for at det tok tid med å få tilgang til alt så har gruppen holdt motivasjon og momentum oppe og jobbet sammen hver dag.

8.3.2 Dokumentasjon

Dokumentasjon har vært en av de største utfordringene gruppen har støtt på, da dokumentasjon er svært viktig, samt grunnlaget for mye av kunnskapshevingen i prosjektet. Det har vært to områder hvor dokumentasjon har vært essensielt, hvor hver av disse har hatt utfordringer. I første omgang, så var det vanskelig å forstå hva Rubrik og dens tjenester er med manglende dokumentasjon. Dette løste gruppen med å ta kontakt med Rubrik selv og ha et møte med Rubriks salgsansvarlig sammen med kontaktperson i Netsecurity. Her fikk gruppen en god gjennomgang og kartlagt tjenesteelementet samt veien videre for hvordan oppsettet skulle være, for eksempel metoden for å hente data fra Rubrik Polaris via API. Etter dette møtte gruppen på en utfordring knyttet til XSOAR sin dokumentasjon. Fordelen med plattformen er at den er basert på åpen kildekode, men alt er ikke like godt dokumentert. Integrasjonene som blir delt er ikke nødvendigvis alltid like godt dokumentert, og kan være alt for spesialisert eller altfor generisk. Det finnes en del “boilerplate”-kode, men en del av kunnskapen er å vite hvor dette skal anvendes. Gruppen var svært oppmerksom på at kommunikasjon med både kontaktperson og teknisk ansvarlig var viktig her, om ikke helt nødvendig. Dette bidro til gruppen og Netsecurity sitt mål om kvalitetssikring, da dette tjenesteelementet skulle videreutvikles, og dermed var dokumentasjon av arbeidet, samt løsningene på utfordringene utrolig viktig.

8.4 Erfaringer og lærdom

Bachelorprosjektet har bidratt til nye erfaringer og et stort læringsutbytte for gruppen. I dette delkapittelet vil gruppen ta for seg noen av de viktigste erfaringene fra prosjektperioden samt hva de har lært.

Gruppen har siden begynnelsen av prosjektet vært klar over at oppgaven ikke kom til å bli enkel, og måtte selv ta styringen for hvordan prosjektet ble driftet og hva som trengs for å få laget tjenesteelementet bedriften etterspurte. Det at gruppen gjennom studieløpet har hatt mye prosjektarbeid og samarbeid har vært en god erfaring å ha med i håndteringen av prosjektet. Denne erfaringen ble videreutviklet da gruppen for det første var nyetablert og ikke hadde jobbet sammen, samt at de valgte å ta i bruk en annen prosjektmetodikk enn hva studiet introduserte for dem. Gruppen har lært at det å starte som ny gruppe med en ny prosjektmetodikk ikke er en dans på roser, men noe som må tilpasses og tilvennes for at dynamikk og flyt i arbeidet skal gå så pent for seg som mulig.

Videre så var det en erfaring i seg selv det å drive med kompetanseheving innenfor områder gruppen ikke hadde vært borti før.
Ingen i gruppen hadde tidligere vært borti GraphQL og når det viste seg å være lite dokumentasjon på Rubrik sitt API så var det utfordrende å tilegne seg kunnskap om hvordan disse skulle knyttes sammen.

Emnene gruppemedlemmene tidligere har hatt undervisning i på universitetet har vært selve grunnpilaren i prosjektet. Spesielt ønsker gruppen å trekke frem kunnskapen og forståelsen de har ervervet gjennom emnene *Samskaping - kommunikasjon og prosjektarbeid*, *Systemanalyse og systemutvikling*, *Tjenestedesign* og *forretningsmodeller* og *Prosjektgjennomføring*, som spesielt relevante for prosjektet gruppen nå har gjennomført. Et eksempel på dette var utfordringen gruppen hadde i starten med å kartlegge prosjektet, ettersom det var mye usikkerhet om hvordan tjenesteelementet hang sammen**.** Gruppen tok i bruk spesifikk kunnskap fra systemanalyse- og utvikling for å lage kartleggingsmodeller. Med dette kunne gruppens analyser og planer formidles til Netsecurity på en god måte.

Avslutningsvis er det verdt å nevne emnet *Praksisprosjekt*, som var slik gruppen fikk kontakt med Netsecurity og hverandre for første gang, og muliggjorde denne lærerike og givende bacheloroppgaven for gruppen.

Gjennom disse emnene har gruppens medlemmer lært å kommunisere godt med hverandre og oppdragsgiver for å sikre god samhandling mellom alle parter, samtidig som forståelsen innen prosjektgjennomføring har gjort det enklere å både se og oppfylle oppdragsgiver sine behov i prosjektet. Gruppen mener erfaringen fra dette prosjektet innen systemutvikling og prosjektgjennomføring legger et godt grunnlag for videre personlig vekst og utvikling innen fagfeltet informasjonssystemer.

Alt i alt har læringskurven gjennom begge delene av prosjektet vært bratt og utfordrende, men gruppen har gjennom dette opparbeidet seg mye kunnskap og erfaring om prosjektstyring via Scrumban og det praktiske arbeidet med GraphQL og Python.

9. Konklusjon

I løpet av bachelorprosjektet har gruppen jobbet med en prosjektbeskrivelse som en praktisk oppgave for implementasjon av et nytt tjenesteelement i bedriftens orkestreringsverktøy. Prosjektet gikk ut på å lage et nytt tjenesteelement; hente kundedata fra backups i Rubrik Polaris som skulle automatiseres for sikker datahåndtering av Netsecurity sine sikkerhetsanalytikere. For å gjennomføre dette ble det laget en integrasjon og playbook som henter data fra Rubrik Polaris og laster den inn som en incident i orkestreringsverktøyet XSOAR.

Gjennomføringen av prosjektet krevde mye innsats fra gruppen, da det var bratt læringskurve i tillegg til introduksjon av ny teknologi og metodikk for prosjektstyring. Det var usikkerhet rundt fremgangsmåten for å oppnå kobling mellom Rubrik Polaris og XSOAR, da Polaris var et nytt tjenesteelement og det skulle komme en ny versjon gruppen var ment å ta i bruk. Forventningen om at en versjon med nye og nyttige funksjoner for prosjektet skulle bli gitt ut tidlig på året var med på å sette visse forventninger til sluttresultatet av tjenesteelementet. I samtaler med Netsecurity ble punkter for det endelige tjenesteelementet utformet. Disse punktene bestod av oppsett og gjennomføring av integrasjon, skript og playbook.

Prosjektgjennomføring- og metodikk var en sentral rolle i gruppens suksess. Ved et fokus på kommunikasjon og fleksibilitet, dannet det fra første dag en standard på kvalitet. Dette førte til et bra og kontinuerlig arbeid ved bruk av Scrumban, da den agile metoden bruker selvbestemte iterasjoner og arbeidsoppgaver for å oppnå målene underveis. Det har vært utfordrende å anvende en ny prosjektmetodikk, ettersom dette ikke har vært pensum, men gruppen har likevel tilnærmet seg god lærdom og erfaring for prosjektarbeid videre.

Avslutningsvis, så er gruppen svært fornøyd med resultatet til tjenesteelementet da det møter de ønskene fra produkteier. Progresjonen har vært god til tross for at prosjektet startet med mye utfordringer og uvitenhet.
Det som bidro til å løse problemene som oppstod var godt samarbeid med Netsecurity og det har ført til et nytt tjenesteelementet som kan taes i bruk, men også videreutvikles.

10. Referanser

Aebersold. K (u.å) *Software Testing Methodologies*. Hentet fra https://smartbear.com/learn/automated-testing/software-testing-methodologies/

Chappell. E (16. juli, 2020) *The Ultimate Guide To Scrumban (2022)* Hentet 8.2.2022 fra https://clickup.com/blog/scrumban/

Digdir.no (u.åa) *Prosjektstyring*. Hentet den 24.mars 2022 fra: https://www.digdir.no/prosjektstyring/prosjektstyring/1417

Digdir.no (u.åb) *Om risiko og risikovurdering*. Hentet den 22.februar 2022 fra https://www.digdir.no/informasjonssikkerhet/om-risiko-og-risikovurdering/3048

Eschwiler, S. (30 mars, 2018) *REST vs. GraphQL*. Hentet 10 mai 2022 fra https://medium.com/codingthesmartway-com-blog/rest-vs-graphql-418eac2e3083

GeeksforGeeks (2020, 21.mai) *Difference between V-model and Waterfall model.* https://www.geeksforgeeks.org/difference-between-v-model-and-waterfall-model/

GeeksforGeeks (2019, 30.september) *Difference between Agile Model and V-Model.* https://www.geeksforgeeks.org/difference-between-agile-model-and-v-model/

IBM Cloud Education (2020, 28.oktober) *Three-Tier Architecture.* https://www.ibm.com/cloud/learn/three-tier-architecture

Kukhnavets. P (14. august, 2019) *Scrumban: How to Combine the Best of Two Agile Methods?* Hentet 8.2.2022 fra https://hygger.io/blog/scrumban-combining-the-best-of-two-methods/

Microsoft (u.å). *Use the Tasks app in Teams*. Hentet den 25. april 2022 fra https://support.microsoft.com/en-us/office/use-the-tasks-app-in-teams-e32639f3-2e07-4b62-9a8c-fd706c12c070

Palo Alto Networks (u.å) *Cortex XSOAR Overview*. Hentet 25.04.2022 fra https://www.paloaltonetworks.com/resources/datasheets/cortex-xsoar-overview

Palo Alto Networks (21. oktober, 2020) *Docker Hardening Guide*. Hentet 11.05.2022 fra https://docs.paloaltonetworks.com/cortex/cortex-xsoar/6-6/cortex-xsoar-admin/docker/docker-hardening-guide

Palo Alto Networks (u.å) *Cortex XSOAR Overview*. Hentet 25.04.2022 fra https://www.paloaltonetworks.com/resources/datasheets/cortex-xsoar-overview

Rubrik, Inc. (2022). *About Rubrik*. Hentet den 25. april 2022 fra https://www.rubrik.com/company

Stoica. M, et al., (2016) *Analyzing Agile Development – from Waterfall Style to Scrumban*. Hentet den 11. mai 2022 fra https://pdfs.semanticscholar.org/84b3/46b298ff706bc55c9dae57bc34798e4ae416.pdf

Stakeholdermap.com (u.å) *Risk Assessment Matrix 4 by 4 example with FREE Download*. Hentet den 24.mars fra https://www.stakeholdermap.com/risk/risk-assessment-matrix-4x4.html

Softwaretestinghelp.com (2022, 3.april) *Functional testing vs non-functional testing*. https://www.softwaretestinghelp.com/functional-testing-vs-non-functional-testing/

Softwaretestinghelp.com (2022, 3.april) *What is integration testing (tutorial with integration testing example)* https://www.softwaretestinghelp.com/what-is-integration-testing/

Unadkat. J (2021, 15.januar) *What is Test Driven Development (TDD) : Approach & Benefits.* https://www.browserstack.com/guide/what-is-test-driven-development

Vertex42 (2021, 12. mars). *Simple Gantt Chart*. Hentet den 25. april 2022 fra https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

11. Vedlegg

Vedlegg 1 – Uttalelse fra prosjekteier

Bacheloroppgave Netsecurity 2022

Netsecurity er en bedrift som satser på unge talenter innenfor IT sikkerhet og IT fagområdene, og har svært gode erfaringer fra både deltidsansatte og praksis-arbeidsplass for studenter på sørlandet. I år har Netsecurity for første gang bachelorprosjekter for studenter ved UiA. Studentene i gruppen har vært i praksis hos Netsecurity høsten 2021, og en av dem er også deltidsansatt på 3. året.

Netsecurity hadde flere tilgjengelige prosjekter og etter samtaler med bachelorgruppen ble oppgaven "Sikker Datahåndtering ved integrering av backuptjeneste" valgt. Oppgaven krever god kjennskap til de aktuelle teknologiene som brukes for å utvikle integrasjonen.

Prosjektet baserer seg på å hente ut relevant informasjon fra eksternt system til Netsecuritys SOC, slik at 24/7 overvåkning av kunders backupsystemer ivaretas. Det er særlig viktig der i en tid der angrep som ransomware blir mer og mer vanlig, at man har kontroll på backupløsning for å sikre man har sikkerhetskopier som er friske. Integrasjonen baserer seg på GraphQL API, og selv om studentene ikke hadde erfaring med dette før prosjektet begynte har de i løpet av semesteret laget en fullverdig integrasjon som kan hente ned og berike alarmer fra eksternt system inn til Netsecuritys hendelseshåndterings-plattform.

Gruppen har hatt en smidig tilnærming til prosjektet med sprinter for stegvis implementering av integrasjonen. Dette har blitt gjort på en ryddig måte med ukentlige møter med Netsecurity for å sikre god fremdrift i prosjektet. Her har det også vært arena for å ta opp spørsmål om utforming av integrasjonen

Studentene har imponert over måten de har satt seg inn i nye teknologier og plattformer på kort tid, som Rubrik Polaris backup, GraphQL API query, XSOAR Playbooking. Studentene har selv gjennomført kurs for flere av disse løsningene for å få en bedre forståelse av hvordan gjennomføre prosjektet på en god måte. De har vist god evne til å forstå problemstillingen og viktigheten av integrasjonen for Netsecurity.

Resultatet av prosjektet har innfridd våre forhåpninger meget bra. Netsecurity har gjennom studentenes kompetansebygging og innsats nå en god plan for å sette i produksjons og leveranse et nytt og ettertraktet element til våre sikkerhetstjenester. Prosjektet har samtidig lagt til rette for en stor effektivisering av våre tjenester innen datahåndtering, og også åpnet for nye markedsmuligheter vi kommer til å jobbe videre med.

Netsecurity vil takke studentene for et vel gjennomført prosjekt og ønsker dem masse lykke til videre!

Frank Kirkeng
Avdelingsleder Secure Operations
Oppgavesponsor

Cato Robstad
Sikkerhetskonsulent
Faglig veileder

Bacheloroppgave UiA 2022 - Netsecurity attest 1

Vedlegg 2. Selvevaluering

Gruppeevaluering

Alt i alt er vi svært fornøyde med hvordan prosjektet har gått og hvordan tjenesteelementet ble. Vi gjorde det tidlig klart for hverandre hva vi forventet av prosjektet, noe som gjorde at vi hadde en god dynamikk gjennom hele perioden. Samarbeidet har fungert svært godt og alle gruppemedlemmene har bidratt i like stor grad. Ved å utnytte hverandres styrker og motivere hverandre underveis, har gjort at vi alle sitter med en god følelse når vi leverer dette prosjektet. Vi er også veldig stolte over hva vi har fått til på denne korte tiden.

Elina W. Antonsen

Det å ha jobbet med et slikt bachelorprosjekt har vært veldig gøy og utrolig lærerikt. Det jeg synes har vært mest interessant er hvordan vi har klart å styre et så flytende prosjekt. Som ansvarlig for tester og kvalitetssikring har jeg hatt hovedansvaret for hvilke tester som det har vært mest relevant å gjennomføre, samt begynne diskusjonen om hva god kvalitet og kvalitetssikring er. Jeg har vært innom områder som har vært både lærerike og utfordrende, blant annet utviklingen av integrasjonen, playbook og de forskjellige testene av tjenesten. Alt dette har gjort at jeg har tilegnet meg mye ny og nyttig kunnskap som jeg vil ta med meg videre inn i ny rolle som testkonsulent i arbeidslivet.

Kjartan Erland

Det har vært utrolig kjekt og utfordrende å jobbe med et så praktisk prosjekt. Å være med å utvikle et tjenesteelement som faktisk skal tas i bruk mot kunder har satt et preg på å gjøre en god jobb. Som hovedansvarlig på det tekniske, å lære seg GraphQL, oppsette av Python i XSOAR og playbooks har ført til et utrolig lærerikt, men også utfordrende bachelorprosjekt.

Det som har vært ekstra spennende er at jeg er deltidsansatt hos Netsecurity som sikkerhetskonsulent, og dermed være med på hvordan tjenesteelementet blir utviklet, har satt jobben i et nytt lys.

Hans Christian Larsen

Det å ha fått muligheten til å arbeide så tett med datasikkerhet har vært en lærerik opplevelse og har gitt meg mange gode erfaringer innenfor et område jeg er interessert i. Det har vært spennende å arbeide med å utvikle et tjenesteelement for en bedrift som faktisk skal ta dette i bruk. Den praktiske kunnskapen jeg har fått gjennom dette prosjektet kommer til god nytte ettersom jeg starter i deltidsstilling hos Netsecurity rundt slutten av prosjektet. Da har jeg allerede fått erfaring i hvordan bedriften og deres systemer henger sammen, i tillegg til å ha et mye bedre utgangspunkt når det kommer til å ta i bruk XSOAR, Python og GraphQL. Når det kommer til prosjektstyring har jeg fått god praktisk erfaring gjennom eksempelvis prosjekstyringsrammeverket vi brukte.

Vedlegg 3. Risikoregister

Skalaen for sannsynlighet er dermed som følger:

Usannsynlig = 1, lite sannsynlig = 2, mulig = 3 og sannsynlig = 4.

For konsekvens ser skalaen noenlunde lik ut: Ubetydelig = 1, moderat = 2, alvorlig = 3 og kritisk = 4.

S = sannsynlighet K = konsekvens T = totalt

Teknisk

Risiko som må taes til etterretning	Begrunnelse	S	K	T
1. Miste tilgang til filer og kontoer	Det å miste tilgang til filer og kontoer vil kunne føre til stans i arbeidet. Prosjektet avhenger av nettopp dette.	1	3	3
2. Miste arbeid pga ingen versjonskontroll	Dette kan være svært kritisk da alt vårt arbeid kan gå tapt. Ved mangel på versjonskontroll mister man mulighet-en til å mellomlagre arbeid og kontroll sikre dataen.	2	3	6
3. Bug i software	Dette kan være kritisk til fremgang i prosjektet, da dette kan gi et annet resultat enn forventet	3	2	6
4. Restriktiv tilgang	Pga konfidensialitet og sikkerhet, så kan det være begrensninger med tilgang via VPN ved hjemmekontor	3	2	6

Personlige/menneskelige

Risiko som må taes til etterretning	Begrunnelse	S	K	T
5. Dobbelt arbeid pga bruk av flere ulike maskiner	På bakgrunn av manglende versjonskontroll så vil man måtte jobbe "lokalt". Gruppen veksler mellom å jobbe digitalt på egne laptoper og dra på kontoret til bedriften og jobbe fra pcene der. Ettersom gruppen ikke har tilgang til det samme på sine laptoper som på bedriftens PC-er, risikerer man fra tid til annen å jobbe dobbelt opp.	3	2	6
6. Sykdom	Vi er på tampen av pandemien, men det er fortsatt en risiko for å bli smittet av covid eller bli syk av noe annet. Det kan føre til at gruppemedlemmer må bli hjemme og kanskje ikke har allmenntilstand til å jobbe med prosjektet.	4	2	8
7. Miste arbeid pga kunnskapsnivå om verktøy	I et prosjekt vil man kunne komme borti verktøy en ikke har brukt før. Når man arbeider med noe nytt og gjør seg kjent med verktøyet så vil det være en risiko for å fjerne noe viktig informasjon.	4	2	8
8. Lite kommunikasjon	Det kan hende det oppstår situasjoner hvor man har lite kommunikasjon med veileder og/eller bedrift	2	3	6