BACHELOROPPGAVE:

MONITORING IS KEY

FORFATTERE:

ØYVIND SIGERSTAD
NILS SLÅEN
BJØRN-ERIK STRAND

Dato:
15.05.2013

SAMMENDRAG

Tittel:	Monitoring is Key	Dato : 15.05.2013
Deltaker(e)/	Øyvind Sigerstad Nils Slåen Bjørn-Erik Strand	
Veileder(e):	Erik Hjelmås	
Evt. oppdragsgiver:	Hedmark fylkeskommune, Serviceenheten IKT	
Stikkord/nøkkelord (3-5 stk)	Overvåkning, Fri programvare, Icinga	
Antall sider/ord: 240	Antall vedlegg: 9	Publiseringsavtale inngått: Ja

Kort beskrivelse av bacheloroppgaven:

En moderne bedrift baserer seg i stor grad på IT-systemer. Dersom det er problemer med disse, går det ofte ut over brukerne av systemene. Derfor er det viktig at de som er ansvarlige for systemene kan varsles med en gang noe er galt, eller er i ferd med å gå galt.

Målet for denne bacheloroppagaven har vært å implementere en overvåkningsløsning for IKT-avdelingen ved Hedmark fylkeskommune, basert på fri programvare. I løsningen er det benyttet i Icinga, som er en fork av Nagios. Videre oppbygging av overåkningsløsnigen består av overvåkning av servere og tjenester, intrastruktur og servermiljø. For oversikt over status for de ulike systeme er det utviklet et statusvindu tilpasset visning på en stor skjerm.

Rapporten går igjennom hvilke valg og vurderinger gruppen har tatt underveis. Dette forklares gjennom en teoretisk bakgrunn for hvordan en setter opp en overvåkningsløsning med Icinga, implementasjonen, sikkerhetsaspekter og dokumentasjon relevant for løsningen.

ABSTRACT

Title:	Monitoring is Key	Date : 15.05.2013
Participants/	Øyvind Sigerstad	
	Nils Slåen	
	Bjørn-Erik Strand	
Supervisor(s)	Erik Hjelmås	
Employer:	Hedmark fylkeskommune, Serviceenheten IKT	
Keywords (3-5)	Monitoring, open source, Icinga	

Number of pages/words: 240	Number of appendix: 9	Availability (open/confidential): Open

Short description of the bachelor thesis:

IT systems are critical to a modern enterprise. Problems with the systems normally affect a great deal of users. It is crucial that the people responsible for the systems are notified of such problems as soon as possible or even before a problem occurs.

The goal of this bachelor's thesis has been to implement a monitoring solution based on open source for the IT department at Hedmark fylkeskommune. For the implementation Icinga, which is a fork of Nagios, was chosen. Further additions to the monitoring solution has been in regards to monitoring servers and services, infrastructure and server room environment. To give a short summary of the current state of all hosts and services a web based tactical overview was developed.

This report discusses which choices and considerations the group has made in regards to the monitoring solution. This is reflected through a theoretical overview of monitoring with Icinga, our implementation, security considerations and documentation relevant for the project.

Forord

Da forslagene til bacheloroppgaver ble presentert, var det ingen oppgaver vi følte utpekte seg som ekstra interessante. Vi ønsket oss en oppgave der vi kunne ha en implementasjon- eller utviklingsdel, der vi fikk bruk for mest mulig av den kunnskapen vi har opparbeidet oss gjennom to og et halvt år ved Høgskolen i Gjøvik.

Vi tok da kontakt med IKT-avdelingen ved Hedmark fylkeskommune for å høre om de hadde noen mulige problemstillinger som kunne passe som en bacheloroppgave. Dette resulterte i en oppgave som passet med våre ønsker, og som vi syntes var meget interessant. Det skulle utvikles et forslag til en ny overvåkningsløsning av servere, infrastruktur og tjenester.

Denne rapporten dekker hvordan vi har gått fram for å fylle flest mulig av avdelingens behov, og hvilke valg og vurderinger vi har tatt underveis.

Gruppen takker:

- Veileder Erik Hjelmås for god hjelp og veiledning gjennom hele prosjektet.
- Svein-Inge Kvalø, Lasse Odden og resten av IKT-avdelingen ved Hedmark fylkeskommune for god kommunikasjon, samarbeid og tilrettelegging for oppgaven.
- Bachelorgruppen dot11ac for bilberging og korrekturlesing.
- Jon Langseth for gjennomgang av overvåkningsløsningen som IT-tjenesten ved HiG bruker.

Gjøvik, 15.05.2013

Øyvind Sigerstad | Nils Slåen | Bjørn-Erik Strand

Ordliste

Daemon En prosess som kjører i bakgrunnen.

Shell Et grensesnitt til et operativsystem funksjoner og tjenester.

SSH Secure Shell. En sikker tilkoblingsprotokoll for å koble til UNIX-shell.

SLA Service-level agreement. En avtale mellom kunde og tilbyder der tjenesten og krav til denne er formelt definert.

I/O Input/Output

Overhead Kontrollinformasjon ved en tilkobling.

LDAP Lightweight Directory Access Protocol. Protokoll for oppslag i en katalogtjeneste.

DNS Domain Name System. Protokoll for oppslag i informasjon om en IP-adresser, deriblant domenenavn.

DHCP Dynamic Host Controll Protocol. Tildeling av IP-adresser for enheter

Ajax Asynkron JavaScript og XML. Metode for å la web-browsere hente data fra web-serveren etter at siden er lastet.

RRD Round-Robin Database. Database med et sirkulært buffer slik at dataene byttes ut over tid og databasestørrelsen er konstant.

API Application Programming Interface. Grenesnitt for kommunikasjon mellom programvare.

GPO Group Policy Object. Sentral administrasjon av konfigurasjon via Microsoft Active Directory.

Fork En utvikler starter en uavhengig og separat programvare utifra eksisterende kildekode.

VPN Virtual Private Network. Punkt-til-punkt-forbindelse som kan være kryptert.

RSS Really Simple Syndication. Benyttes for å abonnere på nettsider via et XML-basert filformat.

Innhold

Tabeller

Figurer

Kapittel 1

Innledning

1.1 Bakgrunn

Serviceenheten IKT ved Hedmark Fylkeskommune (heretter kalt IKT-avdelingen) har det overordnede ansvaret for alt datautstyr som tilhører fylkeskommunen, og datanettverket for sentraladministrasjonen og øvrige lokasjoner. Dette omfatter blant annet videregående skoler, tannklinikker og Hedmark Trafikk. Totalt benyttes IT-løsningene av over 10 000 brukere.

Datasystemene er kritiske for daglig drift av de mange funksjonene fylkeskommunen fyller. Det er derfor mange brukere som berøres dersom en feil skulle inntreffe i systemene. Det er det viktig at de ansvarlige får beskjed så raskt og effektivt som mulig når et system går ned, eller ikke fungerer som det skal.

Per i dag benyttes en enkel overvåkningsløsning, "Servers Alive" [1], som ikke dekker IKT-avdelingens behov. Oppsettet gir for mye irrelevant informasjon, som gjør at en ikke får et godt nok oversiktsbilde. Det mangler også mulighet for å kunne overvåke mange ønskede tjenester. Systemet gir kun varsling til skjerm, men det er i tillegg ønskelig å kunne få varsler på SMS og e-post. Servers Alive er også en proprietær løsning, og dermed ikke lett å utvide. I tillegg til dette kjøres ikke sjekkene parallelt, og derfor skalerer systemet dårlig.

IKT-avdelingen har lenge arbeidet med å finne en bedre overvåkningsløsning. Det er ønskelig å erstatte Servers Alive med en mer modulbasert løsning, der IKT-avdelingen har tilgang til kildekoden og kan utføre endringer. De har imidlertid for hver gjennomgang av mulige overvåkningsløsninger konkludert med at de ikke har de nødvendige ressursene til et slikt prosjekt. Da vi forespurte IKT-avdelingen om de hadde noen mulige bachelorprosjekter, var de veldig interessert i at en bachelorgruppe kunne se på hvordan et slikt prosjekt kunne gjennomføres.

1.2 Oppgavebeskrivelse

Det skal settes opp en overvåkningsløsning, som skal varsle feil og hendelser på enheter i fylkeskommunens datanettverk. Dette innebeærer servere med tjenester, og nettverksenheter som switcher, routere og brannmurer. Dette systemet må være modulært med mulighet for enkelt å kunne legge til nye moduler i ettertid. Følgende krav er satt til løsningen:

- Webgrensesnitt for administering av overvåkningssystemet, der informasjon om enheter og hvilke tjenester som overvåkes skal vises.
- Varsling
  - med feilbeskrivelse.
  - med flere grader kritisk nivå.
  - med mulighet for varsling til SMS, e-post og skjerm.
  - som tar hensyn til redundans.
- Avhengigheter mellom enheter som overvåkes skal kunne settes opp. Varsling skal gi informasjon om det er antatt følgefeil og hovedfeil.
- Objekter (data inn) og data skapt av programvaren (data ut) skal lagres i database(r).
- All tiltstandsinformasjon skal logges og lagres i database.
- WMI, SNMP, ICMP (med antall drop før varsling), og eventuelt andre aktuelle standarder skal støttes.
- Det skal være enkelt å legge til nye enheter for overvåking, fra et grafisk grensesnitt.
- Innen servermiljø skal det kunne overvåkes temperatur, luftfuktighet og UPS.

Vi må derfor finne programvare som kan tilpasses fylkeskommunens systemer, og som dekker flest mulig av disse kravene. Dersom eksisterende løsninger ikke finnes, eller ikke dekker behovet, må vi utvikle dette selv.

IKT-avdelingen har også en liste over tilleggsfunksjonalitet de ønsker implementert i løsningen. Disse vil implementeres dersom det er tid til det.

- Rapporter som viser dagens situasjon og trender.
- Ta hensyn til driftsarbeid. Planlagt arbeid definert på enheter vil ikke utløse alarm, men driftsarbeidsvarsling.
- SLA
  - Vise SLA-avtaler opp mot faktiske verdier.
  - Kundebase med krav opp mot logget data.
  - Kapasitetsmålinger opp mot maksverdier (for eksempel linjekapasitet).
- Dashboard-funksjonalitet med mulighet for personliggjøring av grafisk grensesnitt.
- Grensesnitt for mobile enheter.
- Konfigurasjonskart som tegner kart av løsning grafisk med klikkbare objekter (enheter).
- Mulighet for lagring av konfigurasjonsoppsett for objekter med integrasjon mot CMDB.
- Overvåking av telefonlinjer via Trio og Asterisk.

Etter at valg av kjerneprogramvare var tatt ble det definert at følgende kategorier med underpunkter skulle overvåkes:

- Infrastruktur og nettverk.
  - Brannmurer.
  - Switcher.
  - Trådløse kontrollere.
  - VMware ESX og Citrix Xen.
  - Serverrommiljø.
    - Temperatur.
    - Strøm (UPS).
    - Luftfuktighet.
- Servere
  - CPU.
  - Minne.
  - Disk.
  - Prosesser.
  - Applikasjoner.
  - Databaser: MySQL, MySQL Cluster, Oracle og MSSQL.
  - E-postserver: Microsoft Exchange.
  - Tjenester.
    - Webservere.
    - DNS.
    - DHCP.
    - LDAP.

1.3 Avgrensninger

Fordi oppdragsgiver har både krav og ønsket tilleggsfunksjonalitet, må oppgaven avgrenses slik at tidsrammer for å oppfyllene kravene overholdes. Dersom vi ligger foran tidsskjemaet, vil vi begynne å se på og implementere tilleggsfunksjonalitet.

Manuell endring av konfigurasjon i tekstfiler er den mest fleksible løsningen, og oppdragsgiver var enig i at dette var enkelt nok og dermed ikke trengte å være grafisk.

Kravet til at alle data inn og ut skal logges i databaser bortfaller, da dette gjennom testing genererte for store datamengder til at det var hensiktsmessig å lagre alt.

1.4 Målgruppe

Overvåkningsløsningen utvikles for IKT-avdelingen, og ansatte her vil være en av målgruppene. Selve rapporten vil vinkles slik at medstudenter, veileder, oppdragsgiver, sensor og ellers andre interesserte kan få et innblikk i hvordan prosessen med å implementere en overvåkningsløsning basert på fri programvare har vært. Det faglige nivået på rapporten vil være slik at medstudenter kan få utbytte av stoffet.

1.5 Prosjektmål

Effektmål

I forhold til dagens løsning skal den nye løsningen

- være mer oversiktlig.
- tilrettelegge for mer omfattende overvåkning.
- være mer fleksibel.
- bidra til mer effektiv drift for IKT-avdelingen, der hendelser raskt kan oppdages og eskaleres.

Resultatmål

- Utvikle en overvåkningsløsning som tilfredsstiller de krav som er satt.
- Den nye overvåkningsløsningen vil resultere i at den eksisterende kan erstattes og deretter fases ut.

Læringsmål

- Sette oss inn i "best practices" for overvåking av datasystemer.
- Kunne tilpasse og implementere en overvåkningsløsning for små og mellomstore bedrifter.

1.6 Rammer

Følgende krav var gitt i oppgaven

- Dersom eksisterende programvare benyttes må denne være fri programvare.
- MySQL skal benyttes som databasesystem.
- Prosjektrapporten skal ikke inneholde sensitiv informasjon om Hedmark fylkeskommunes tekniske løsninger. Eksempelvis IP-adresser, brukernavn/passord eller brannmurkonfigurasjon. Dette er i tråd med taushetserklæringen som må inngås.
- Det skal kjøpes inn en enhet for å overvåke luftfuktighet og temperaturer. Innkjøp av denne skal skje i henhold til fylkeskommunens innkjøpsrutiner.

1.7 Faglig bakgrunn

Bjørn-Erik og Øyvind er 3. års studenter på Drift av nettverk og datasystemer, mens Nils går 3. året på Dataingeniør. Gjennom to og et halvt år på Høgskolen i Gjøvik har vi tatt fag innenfor områdene programmering, nettverk, databaser, sikkerhet, statistikk og ulik grad av matematikk. Nils og Øyvind har tidligere vært lærlinger ved henholdsvis Hedmark fylkeskommune og Hamar katedralskole, der de fikk kunnskap om deres applikasjoner og infrastruktur.

Bjørn-Erik og Øyvind har også tidligere skrevet en rapport om bruk av ITIL ved Hedmark fylkeskommunes IKT-avdeling, i faget "IT Service Management".

Vi mener at en overvåkningsoppgave med denne beskrivelsen kan dekke mange av områdene vi har vært gjennom i løpet av studiet. Her er det også områder der gruppen har delvis eller manglende kunnskap, både om hvilke teknologier som finnes og hvordan de brukes i et større datamiljø.

1.8 Roller

Prosjektleder

Prosjektleder er Øyvind Sigerstad. Denne rollen vil være fast under hele prosjektet. Lederens hovedansvar er å se nødvendigheten for møter, samt ha et overordnet ansvar for arbeidsfordelingen. Prosjektlederen vil også ta endelige avgjørelser dersom gruppen ikke kan komme til enighet, som stipulert i gruppereglene.

Kommunikasjonsansvarlig

Vår kontaktperson er Nils Slåen. All kontakt med oppdragsgiver og annen utgående kommunikasjon vil gå via ham. Kommunikasjonsansvarlig har også ansvar for å avtale møter.

Webansvarlig

Vår webansvarlig er Bjørn-Erik Strand. Hans hovedansvar er å sette websiden vår i drift før fristen. Webansvarlig har også et overordnet ansvar for å publisere oppdateringer. De andre medlemmene i gruppen skal også ha tilgang til å legge ut informasjon.

Oppdragsgiver

Vår oppdragsgiver er Svein-Inge Kvalø (Fungerende driftsleder ved IKT-avdelingen, Hedmark fylkeskommune), og er vårt kontaktpunkt ved Hedmark fylkeskommune. Lasse Odden er vår tekniske kontakt (IT-konsulent ved IKT-avdelingen, Hedmark fylkeskommune). Han vil være en ressurs som brukes for tekniske spørmål rundt IKT-avdelingens oppsett.

Veileder

Erik Hjelmås (Førsteamanuensis, Dr. scient ved Høgskolen i Gjøvik), er vår veileder gjennom bacheloroppgaven. Han vil bidra til avklaringer rundt oppgaven og komme med innspill rundt tekniske utfordringer.

1.9 Rutiner og regler i gruppa

Et eget dokument er utarbeidet med grupperegler som alle gruppens medlemmer er enige i og har skrevet under på. Dette finnes i Vedlegg G side 14.

1.10 Verktøy

Alt av egenutviklet kildekode og konfigurasjonsfiler vil bli lagt under et SVN repository. Dette for å få versjonskontroll og samle alt på et sted, som vi kan ta backup av. Google Docs vil bli brukt for å samarbeide på dokumenter. Når den endelige rapporten skal genereres, vil vi overføre Google Docs-dokumentet til LaTeX. Dropbox vil bli benyttet til å dele filer og dokumenter innad i prosjektgruppen, og med oppdragsgiver. Vi vil bruke Wordpress som publiseringsverktøy for hjemmesiden.

Av andre verktøy skal Microsoft Visio benyttes til å lage illustrasjoner og matplotlib til diagrammer.

1.11 Utviklingsmodell/rammeverk

For å tilpasse overvåkningssystemet til IKT-avdelingens krav, vil det med stor sannsynlighet komme endringer etterhvert. Det vil kunne oppstå ulike situasjoner som er vanskelige å forutse. Eksempler på dette er at en plugin som kreves ikke eksisterer, eller at webgrensesnittet må endres på grunn av mer informasjon eller omorganisering. I tillegg kan oppdragsgiver komme med innspill som fører til endring av funksjonalitet. Derfor må vi ha en fleksibel modell med tanke på endringer og tilbakefall, men med begrensninger på hvor mye tid vi kan bruke ved uforutsette problemer.

På grunn av en naturlig sammenheng og avhengigheter mellom kravene som er satt, har vi valgt å gruppere funksjonalitet i moduler. De forskjellige modulene er "Kjerneprogramvare", "Server", "Infrastruktur", "Servermiljø", "Varsling" og "Statusvindu" (visualisert i Vedlegg G side 12). Med nærliggende sammenheng mener vi at funksjonalitet som CPU, RAM, disk og prosesser er i samme kategori.

Planen er også å gjennomføre kategoriene i en gitt rekkefølge. Varsling kan settes opp uten noe input, men med tanke på redundans i systemene, avhengigheter mellom objektene, og testing av funksjonene som går inn under varsling, ser vi på dette som en avhengighet av Server og Infrastruktur. Statusvindu har vi valgt å legge sist fordi vi ser for oss at modulene Server, Infrastruktur, Servermiljø, og Varsling må være på plass før en varslingsskjerm kan ferdigstilles.

I tradisjonell systemutvikling må en definere alle parametere og rekkefølge på alt som skal utvikles, før en starter med selve utviklingen. En smidig modell vil gi oss mer frihet til å gjøre justeringer underveis. Vi har derfor valgt en iterativ modell [2]. En del av den iterative modellen er "product control list", men denne har vi valgt å ikke ta med fordi vi har satt opp en overordnet kategorisering av funksjonalitet, og bestemt når hver av disse skal implementeres.

Vi vil dele inn iterasjonene i faser, som hver består av en planleggingsfase, en implementeringsfase, og en fase for testing og evaluering. Disse vil så gjentas for hver modul som skal implementeres. Halvveis i hver iterasjon vil vi ha et møte med oppdragsgiver der vi kan presentere funksjonalitet i modulen, og få tilbakemeldinger. Det gjør at vi kan jobbe med dette i den siste halvdelen av iterasjonen. Dette vil forhåpentligvis føre til at produktet ikke sporer av fra de forventningene og behovene IKT-avdelingen har.

1.12 Om rapporten

1.12.1 Oversikt over kapitler

1. Innleding - oppgaven inntroduseres. Bakgrunn, formål og plan for gjennomføring gjennomgås.
2. Teori - en teoretisk bakgrunn innen overvåkning og konfigurasjon av overvåkningsløsningen.
3. Implementasjon - gjennomgang av implementasjonen av overvåkningsløsningen.
4. Sikkerhet - de sikkerhetsvurderingene gruppen har tatt i implementasjonen presenteres.
5. Dokumentasjon - rutiner og informasjon gruppen anser som nyttig tilleggsinformasjon til implementasjonen.
6. Avslutning - rapporten avsluttes og konkluderes.

1.12.2 Konvensjoner

Kildekode vil inneholde "syntax-highlighting" til riktig programmeringsspråk og linjenummer. Dette markeres slik:

```
1 #!/usr/bin/perl
2 print "Hello World.\n";
```

Konfigurasjonsfiler og kommandoer markeres slik:

```
define command {
    command_name    check_ssh
    command_line    $USER1$/check_ssh -H $HOSTADDRESS$ -p $_HOSTSSHPORT$
}
```

En linje i en konfigurasjonsfil omtales som et "direktiv". En spesifikk verdi omtales som et "attributt" (for eksempel command_name). Der kun utdrag av konfigurasjon eller kildekode er benyttet brukes "..." for å markere hvor det er utelatt.

Objekttyper omtegnes med det navnet som er brukt i dokumentasjonen for eksempel "host", "service", "timeperiod". "Tjeneste" brukes om applikasjoner påpå en server, som benyttes av andre enheter eller brukere. "Host" eller "enhet" brukes om en enhet med en IP-adresse.

1.12.3 Grafikk

Grafene i rapporten er generert av Graphite. Alle illustrasjoner er enten egenprodusert eller hentet fra Icingas dokumentasjon, der de er publisert under GPLv2.

Kapittel 2

Teori

I dette kapittelet gjennomgås teori og konsepter som er nødvendig å forstå før implementasjonsdelen. Alternativer til kjerneprogramvare presenteres og valg av arkitektur blir gjennomgått.

2.1 Hvorfor overvåke

Nesten alle bedrifter i dag bruker IT-systemer for å gjøre seg mer effektiv og konkurransedyktig i markedet. Mange av disse systemene blir ofte tatt for gitt, fordi de rett og slett bare fungerer. Men hva skjer om e-posten en dag ikke virker, eller månedens lønn ikke kommer når den skal? I slike tilfeller kan en overvåkningsløsning være viktig for å oppdage og identifisere problemet tidlig.

Der en har avtaler knyttet til IT-tjenestene som blir levert, kan et overvåkningssystem være med på å dokumentere at tjenestene har vært i henhold til SLA. Overvåkning kan også bidra til å gi et bedre bilde på hvor pålitelige systemer og utstyr er, og kan hjelpe til med å kartlegge hvor systemene er sårbare. Dette kan igjen benyttes ved kapasistetsplanlegging. For de ansatte ved en driftsavdeling kan det også skape en trygghetsfølelse å vite at systemer overvåkes, og feil vil varsles.

Når en leverer tjenester innenfor IT er det viktig at det ikke er kunden som varsler om feilen. Da ender kundene opp som overvåkningsmekanismen. Kriteriene for varsling må være finjustert, slik at en kan reagere før systemene krasjer.

"If you aren't monitoring it, you aren't managing it."

— Limoncelli, Hogan & Chalup , *The Practice of System and Network Administration [3]*

2.2 Generelt om overvåkning

Overvåkning deles ofte inn i kategoriene historisk overvåkning og sanntidsovervåkning. Ved historisk overvåkning blir data samlet inn for å ha statistikk om oppetid, ytelse og bruk. Ut fra disse dataene kan en trekke konklusjoner, som for eksempel at VPN-tjenesten har hatt en oppetid på 99,9 % over det siste året og at gjennomsnittlig antall brukere har vært 100, med en økning på 10 % den siste måneden. Dermed kan en få et overblikk over hvor mye nedetid tjenesten har hatt, og en har dokumentasjon på hvordan leveransen har vært opp mot avtalt leveranse. [3]

Ved å se på stigningstallet kan en også forsøke å si noe om trenden og forventet fremtidig nivå, som kan gi en indikasjon på nødvendige skaleringstiltak. For eksempel at med dagens bruk av toner på kopimaskinen, vil den gå tom for toner innen fredag.

Sanntidsovervåkning vil si at en sjekker tilstanden til en enhet eller tjeneste i et gitt intervall, og så fort noe avviker fra normalen sendes det ut varsel. Målet er at de ansvarlige skal kunne vite om noe er feil så fort som mulig, og helst før brukerne.

2.2.1 Overvåkning i ITIL

ITIL (Information Technology Infrastucture Library) er et sett med standarder og konsepter som beskriver hvordan IT-tjenester kan kvalitetsikres innenfor leveranse, drift og support. ITIL brukes i dag av IT-bedrifter over hele verden og gir et felles rammeverk for hvordan en bør praktisere innen IT-sektoren. ITIL blir definert i et sett med bøker som blir utgitt av Office of Government Commerce i Storbritannia.

Innenfor ITIL er involvering av ulike prosesser ved hendelser essensielt. Overvåkning av systemer og infrastruktur bak forretningskritiske systemer, og overvåkning av forretningskritiske systemer i seg selv er viktig i ITIL-prosessen Service Operations. Overvåkning og varsling er et viktig ledd for kunne å registrere en hendelse som kan være et engangstilfelle, eller en indikasjon på et underliggende problem. Et eksempel her vil være om en bruker melder inn at det er problem med innlogging på en terminalserver. Her kan et engangstilfelle være at passordet har løpt ut for denne brukeren, eller indikere et underliggende problem med LDAP-autentisering.

Når en hendelse inntreffer, enten ved at en bruker rapporterer om en feil eller at overvåkningsløsningen varsler om det, vil servicedesk være første instans som må reagere. Servicedesk er derfor et viktig ledd for å opprettholde forretningskritisk funksjonalitet, og en god statusvisning og tilrettelagt varsling vil bidra til å raskt identifisere hendelsen.

Overvåkning er også en del av komponenten "Capacity Management" innenfor kategorien "Service Delivery", som legger retningslinjer for proaktive handlinger framfor reaktive. Proaktive handlinger innebærer å være i forkant av hendelser som kan oppstå, og ikke reagere når det allerede har skjedd; reaktivt. Informasjon fra overvåkningen kan hjelpe til med å identfisere situasjoner som kan håndteres før hendelser inntreffer og får innvirkning for brukere. [4–6]

2.2.2 Hvordan overvåke

Den enkleste formen for overvåkning av en server er å sende en ICMP echo request (ping), og se om en får et svar (også kjent som pong). En måler tiden til en får et svar tilbake, og sjekker hvor mange prosent av antall forsøk en får svar på. Dette betyr at enheten som kontaktes har en delvis fungerende IP-stack og om linjen imellom fungerer. Men det forteller ikke noe om tjenestene som kjører. Det er tross alt tjenestene som kjører på enheten som er grunnen til at en trenger enheten i utgangspunktet.

En bedre løsning er å teste hver av tjenestene. For en webserver kan det være å prøve å hente ned forsiden av hjemmesiden som skal leveres og sjekke at en får reponsen "HTTP OK 200". Da vil en vite at webserveren kjører og leverer ut sider riktig. Et skritt videre kan være å sjekke at siden inneholder riktig informasjon og at det ikke har blitt endret av uvedkommende (defacing).

2.3 Hvilke overvåkingsløsninger finnes

Det finnes i dag mange løsninger for overvåkning av IT-systemer [7]. Det ble tatt utgangspunkt i prosjektets rammer og krav til funksjonalitet, og valgt ut noen løsninger som ble undersøkt nærmere. Punktene om ønsket tilleggsfunksjonalitet ble også vektlagt slik at de skulle være lettere å implementere i senere faser av prosjektetet, eller i senere prosjekter.

Det ble også gjort en gjennomgang med oppdragsgiver, hvor de ulike løsningene ble diskutert (presentasjon av dette ligger i Vedlegg C). De overvåkningsløsningene som ble vurdert nærmere etter den første runden med research er alle fri programvare og har en utbredt brukermasse.

Zenoss har en open source-lisens på Zenoss Core. Zenoss brukes av mange store aktører [8], som gjorde denne til en mulig kandidat. Det ble likevel valgt å ikke bruke Zenoss på grunn av vinklingen de har mot "Enterprise Solutions" [9]. Dette involverer at en må kjøpe tilleggsmoduler til selve kjernen for å oppnå en komplett overvåkningsløsning.

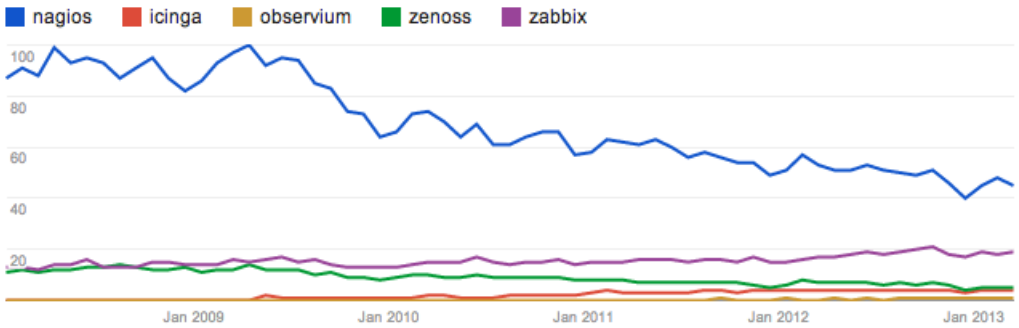
Observium har lagt mye vekt på å visualisere ytelsesdata. Dette vises med statistikk og grafer over hver enhet som overvåkes. Observium erstatter ikke en såkalt "UP/Down overvåkning", da alarmer og grenseverdier for tjenester og servere ikke støttes [10]. Dette gjør at Observium ikke kunne benyttes i denne oppgaven.

Zabbix [11] var en av de sterkeste kandidatene. Denne støtter mange av punktene i oppgavebeskrivelsen, men antall plugin-er utenfor selve programvaren er for få. Zabbix er også et mer monolittisk system, og ikke like modulært som andre kandidater [12]. Oppgavebeskrivelsen krever modularitet, og Zabbix ble derfor valgt bort.

Nagios er kjent som en av de mest utprøvde overvåkningsløsningene [13–16], og har et stort antall brukere som aktivt bidrar til å utvikle plugins og moduler til Nagios-kjernen [17]. Det finnes flere websider der brukere bidrar med og diskuterer plugin-er og moduler, blant annet Nagios Exchange [18] og Monitoring Exchange [19]. Det som gjør Nagios spesielt interessant er fleksibiliteten med tanke på konfigurasjon, integrasjon og plugin-er.

Icinga er en fork av Nagios, og har blitt utviklet videre siden 2004. Konfigurasjoner, plugin-er og addon-er er kompatible med det som benyttes for Nagios. Forskjellen er selve arkitekturen som i Icinga består av tre separate deler, med abstraksjon mot databaselaget.

I Figur 2.1 vises en oversikt over hvor ofte hver av disse overvåkningsløsningene har blitt benyttet i et Google-søk i perioden januar 2008 - januar 2013. Dette er hentet fra Google Trends og viser at Nagios har vært, og fortsatt er et mer populært søkeord enn de andre løsningene.



Figur 2.1: Google Trends: overvåkningsløsninger

2.4 Valg av kjerneprogramvare

Valg av kjerneprogramvare var gruppens viktigste valg, da det la grunnmuren for resten av prosjektet, og ville få stor innvirkning på sluttresultatet. Av kandidatene nevnt i 2.3, ble Icinga og Nagios videre vurdert.

Icingas distribuerte system består av kjernen, som står for prossessering, et databaselag, og administrering av systemet via et webgrensesnitt. Dette er tilrettelagt for et distribuert oppsett. Nagios sin arkitektur har avhengigheter mellom web og lagring av status, som fører til at disse ikke kan separeres. Det er derimot mulig å separere ut MySQL-databasen som Nagios benytter [20].

For uthenting av informasjon i Icinga er det utviklet et API som brukes via Icinga Web. Dette gir ett punkt for uthenting av data, med samme oppbygging for alle spørringer. I Nagios er det ikke noen standardisert måte å hente ut informasjon [21].

En annen fordel med Icinga er støtte for LDAP-autentisering i Icinga Web, som videre kan benyttes til å styre tilgang til ulik informasjon for brukere. Med Nagios kan en også innføre LDAP-støtte, men dette blir da autentisering via en web-server, og videre kontroll av aksess er ikke mulig. Nagios har noe aksesstyring, men det er bare på et objekt kalt "contactgroups". Icinga har aksesstyring på en rekke flere objekter [22].

For databaseoppsett støtter Icinga de tre databasemotorene PostgreSQL, Oracle og MySQL. Nagios har bare støtte for MySQL. I oppgavebeskrivelsen er det krav om MySQL, men for fremtidig bruk av Icinga er det mulighet for å benytte de to andre.

Icinga har også en tilleggsmodul kalt "Icinga Mobile". Denne gir tilrettelagt grensesnitt for mobile enheter, som var ønsket i oppgavebeskrivelsen. Icinga Mobile kan brukes for å gi et grensesnitt tilpasset mindre skjermer. Ved utsendelse av SMS eller e-post til en eller flere ansatte, kan Icinga-Mobile brukes via samme enhet for å se mer utfyllende informasjon om problemet, samt gjøre administrative handlinger.

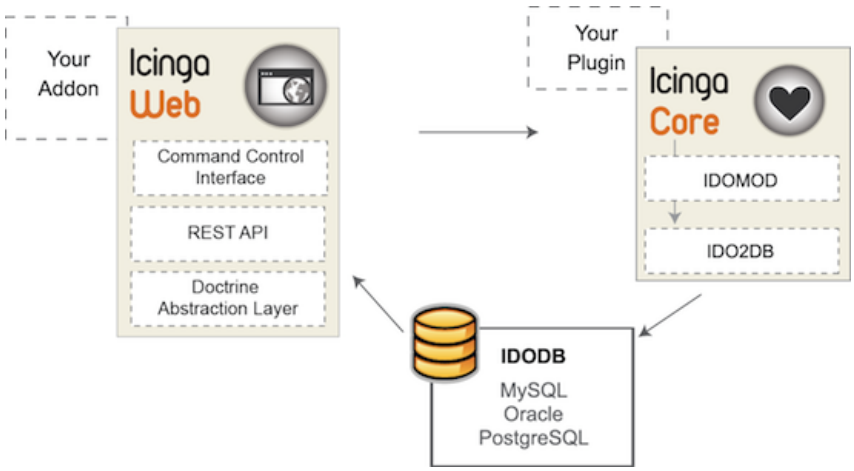
Etter en vurdering av denne funksjonaliteten, ble det besluttet å basere overvåkningsløsningen på Icinga framfor Nagios.

2.5 Icinga

For å kunne forstå implementasjonsdelen av prosjektrapporten er det behov for noe innsikt i hvordan Icinga fungerer. I dette delkapittelet er det viktigste forklart.

Arkitektur

Icinga består av tre separate komponenter, som vist i Figur 2.2.



Figur 2.2: Icingas tre komponenter

Core

Icinga Core håndterer planlegging av sjekker gjennom plugin-er og tar i mot resultatene av disse. Denne informasjonen sendes gjennom SSL-krypterte TCP-sockets videre til IDO2DB-prosessen (Icinga Data Out to Database) gjennom et interface kalt IDOMOD (Icinga Data Out Module). Ved å bruke disse abstraksjonslagene kan databasemotoren enkelt byttes ut. I dokumentasjonen er det laget veiledninger for MySQL, PostgreSQL og Oracle [20].

IDOMOD og IDO2DB kommer i en samlet pakke (IDOutils), men kan separeres ut for å skape et distribuert oppsett. CERN i Sveits har nylig implementert et slikt oppsett med hjelp av lastbalansereren "Mod Gearman" [23].

Web

I utgangspunktet kommer ikke Icinga med noe webgrensesnitt. Men det er mulig å installere to forskjellige pakker for å få dette. I begge kan en få en oversikt over tilstanden til enheter og tjenester, se konfigurasjon og utsendte varsler, samt utføre administrative oppgaver. Det kan også hentes ut statistikk over oppe- og nedetid.

Icinga Classic baserer seg på det samme vindusoppsettet som Nagios. For uthenting av data benyttes CGI-moduler som henter ut data fra filen ”status.dat”. Filen blir brukt av Icinga for å lagre tilstandsinformasjon om enheter og tjenester, kommentarer, og informasjon om nedetid.

Icinga Web er en total omskrivning av webgrensesnittet. Det er en Ajax-drevet, Web 2.0-inspirert front-end, som har flere lag mellom kjernen av Icinga og visningene:

- **Doctrine Abstraction Layer** (DAL) henter informasjon fra databasen. Det kan også benyttes av utviklere for å legge til egne moduler i Icinga Web med uthenting av informasjon fra andre databaser.
- **REST API** presenterer dataene fra DAL til Icinga Web. Her kan autentisering utføres slik at en kan sette hvilken informasjon ulike brukere skal kunne se.
- **Command Control Interface** kan videresende kommandoer til Icinga Core. For eksempel manuell kjøring av en sjekk eller restart av Icinga.

2.6 Objekter

All konfigurasjon av Icinga gjøres i tekstfiler. I selve konfigurasjonsfilen “icinga.cfg” settes filbanen til konfigurasjonsfilene. Icinga vil da ved oppstart lete rekursivt gjennom mappen etter filer som ender med “.cfg”.

Konfigurasjonsdataene er bygget rundt det som i Icinga kalles objekter. Dette er en samling av konfigurasjon som hører sammen. Selv om en ikke kan si at Icingas objekter er det samme som objekter i programmeringsverden, benyttes mange av de samme begrepene og konseptene.

En rekke objekter er definert i Icinga, som vist i Tabell 2.1.

I tillegg til disse kan host-, service- og contact-objekter grupperes med henholdsvis hostgroup, servicegroup og contactgroup.

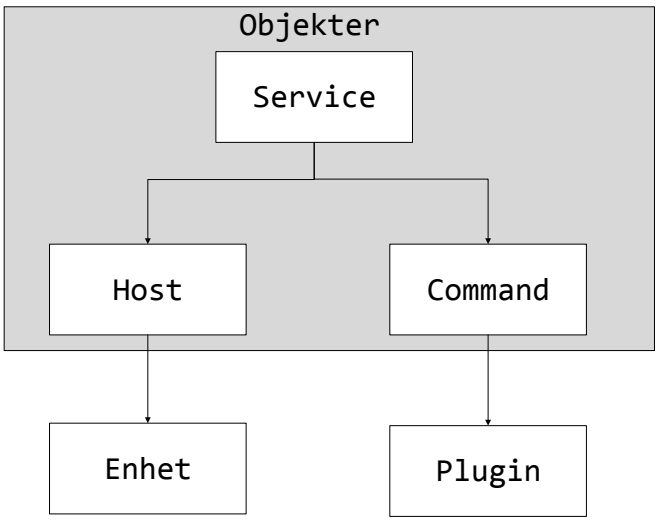
Det er også to objekter med metainformasjon, “hostextinfo” og “serviceextinfo”, der en kan definere ekstra konfigurasjon som bildebaner, notater og koordinater for host- og service-objekter.

Den siste objekttypen er “module”. Her spesifiseres konfigurasjon for en modul som utvider funksjonalitet i Icinga Web.

Sammenhengen mellom objektene er viktig å forstå for å skjønne hvordan sjekker utføres i Icinga. I et service-objekt defineres hvilket command-objekt som skal benyttes for å teste tjenesten og hvilke host- eller hostgroup-objekter dette skal sjekkes på. Dette er vist i Figur 2.3.

Objekt	**Forklaring**	**Eksempel**
Host	En enhet med en adresse IP-adresse.	Server, switch eller router.
Command	En kommando som skal kjøres på overvåkningsserveren.	Et program, for eksempel check_ping.
Service	Kombinasjonen av et host-objekt og et command-objekt.	Sjekk oppetid: kommandoen "check_uptime" skal kjøres på host-objektet"dc1".
Servicegroup	Service-objekt(er) knyttet til host-objekt(er) samles.	En applikasjon har tre prosesser som må være kjørende for at applikasjonen fungerer. Disse grupperes under samme servicegroup.
Contact	Når og hvordan en person skal kontaktes angående et service-objekt.	Jens skal få en SMS hvis DHCP ikke er tilgjengelig.
Timeperiod	Navn og definisjon av en tidsperiode.	Mellom 08.00 og 16.00 på annenhver tirsdag, hvis det er den tredje dagen i måneden.
Host dependency	En eller flere avhengigheter mellom host-objekter.	kantswitch1 er avhengig av kjerneswitch.
Service dependency	Samme som host dependency, men for service-objekter.	Captive portal er avhengig av RADIUS.
Host escalation	Hva skal skje etter at en host har vært nede etter en definert tid.	Om dc1 har vært nede i 30 minutter, send e-post til Ola Sysadmin. Etter 60 minutter sendes SMS til alle i avdelingen
Service escalation	Samme som host escalation, men for service-objeker.	Om DNS har vært nede i 30 minutter, send e-post til Kari Sysadmin. Etter 60 minutter sendes SMS til alle i avdelingen.

Tabell 2.1: Oversikt over objekter i Icinga



Figur 2.3: Sammenhengen mellom host-, service- og command-objekter.

I eksempelet under vises konfigurasjonen for å sette opp en ping-sjekk på HiG1.

```
define host {
    use                     generic_host
    host_name               HiG1
    address                 192.168.0.1
}

define service {
    use                     generic_service
    host_name               HiG1
    service_description     Ping
    check_command           check_ping!200,5%!500,10%
    check_interval          5
}

define command {
    command_name            check_ping
    command_line            /usr/lib/nagios/plugins/check_ping -H $HOSTADDRESS$
        --warning $ARG1$ --critical $ARG2$
}
```

2.7 Plugin-er

Icinga i seg selv kommer uten noen overvåkningssjekker. Til dette benyttes plugin-er.

En plugin i Icinga er et eksternt program som kjøres og outputen hentes inn som data. Resultatet av sjekken hentes fra exit-koden [24] og oversettes til en tilstand, som vist i Tabell 2.2.

I eksempelet under vises en enkel plugin der exit-koden til en ping-kommando sjekkes. Hvis ping ikke får svar, vil den gi exit-koden 1. Scriptet vil da gi exit koden 2, som i Icinga oversettes til tilstanden "CRITICAL".

```
#!/usr/bin/env bash
loss=$(ping -c 5 google.com)
if [ $? -eq 0 ]; then
    echo "OK"
    exit 0
else
    echo "ERROR"
    exit 2
fi
```

Det finnes et sett med offisielle plugin-er til Nagios; nagios-plugins, som utgis av "Nagios Plugins project". Pakken finnes i en rekke pakkebehandlere og inneholder over 50 plugin-er som er ment for å dekke et grunnleggende overvåkningsbehov [25].

Returkode fra plugin	**Service-tilstand**	**Host-tilstand**
0	OK	UP
1	WARNING	UP eller DOWN/UNREACHABLE*
2	CRITICAL	DOWN/UNREACHABLE
3	UNKNOWN	DOWN/UNREACHABLE

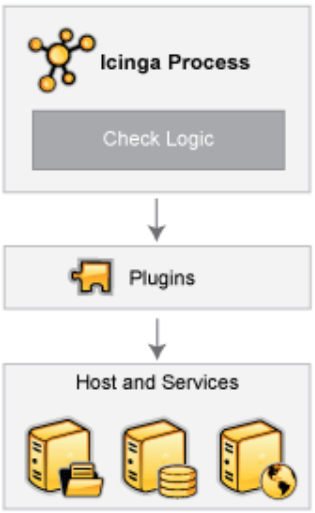
*Ved bruk av "Aggressive host checking" [26].

Tabell 2.2: Tilstandsoversikt

2.8 Sjekker

Tilstanden for et service- eller host-objekt vil bestemmes ut i fra et returnert resultat fra en aktiv eller passiv sjekk. Forskjellen på disse to vil bli forklart i de neste delkapitlene.

Dersom en aktiv eller passiv sjekk resulterer i en annen tilstand enn "OK", er det to typer av tilstanden, "SOFT" og "HARD". Første gang en plugin returnerer en feil vil tilstanden bli satt til SOFT, i tillegg til service- eller host-tilstanden. Etter dette vil sjekken kjøres hyppigere for å finne ut om det er snakk om et forbigående problem. Etter et gitt antall ganger der plugin-en fortsatt ikke returnerer OK, vil tilstanden skifte til HARD og det sendes ut varslingsmelding. Varsling er beskrevet grundigere i 3.12.



Figur 2.4: Aktive Sjekker

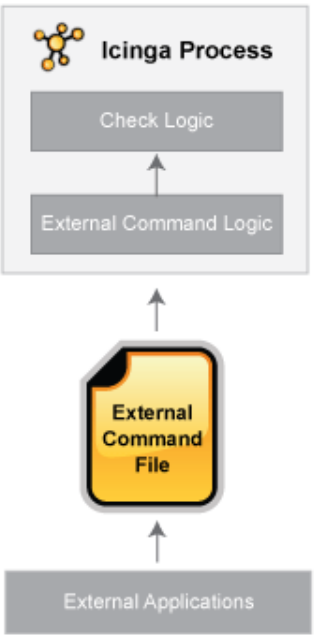
Aktive sjekker

En aktiv sjekk er en sjekk som initieres av Icinga, der en kommando kjøres på Icinga-serveren. Dette er også kjent som en "poll"-modell. En aktiv sjekk defineres ut i fra et command-objekt med parametere og et tidsintervall i service- og host-objekter. I command-objektet defineres en plugin som skal kjøre, og returverdien fra denne vil bestemme hvilken tilstand service- eller host-objektet har. I Figur 2.4 vises komponentene involvert i en aktiv sjekk.

Passive sjekker

Passive sjekker går motsatt vei av aktive sjekker. Her vil hver host si ifra når verdier har overgått en satt grense eller en feil har oppstått. Dette kalles ofte for en "push"-modell. Passive sjekker kjøres ved at et eksternt program skriver en linje med informasjon om tjenesten og sjekkresultat til en fil, som Icinga sjekker periodisk. Dette er illustrert i Figur 2.5.

Fordelen med passive sjekker er at Icinga vil oppdage feil raskere. Ved standard konfigurasjon sjekkes filen med sjekkresultater så ofte som mulig, mens aktive sjekker kjøres hver 5. minutt. Passive sjekker vil også avlaste Icinga-serveren, da alle sjekker utelukkende vil bli kjørt lokalt på enheten som overvåkes. Ulempene med aktive sjekker er at en må åpne for at kommunikasjon kan initieres fra alle enheter som skal overvåkes til Icinga-serveren og at en mister muligheten til å samle all konfigurasjon på ett sentralt sted.



Figur 2.5: Passive Sjekker

Hostsjekk

Resultatet av en host-sjekk vil avgjøre om tilstanden til host-objektet er "UP" eller "DOWN".

Det er ikke et fast tidsintervall for en hostsjekk med mindre check_interval er definert for objektet. Icinga vil kjøre sjekken service-objektet som er satt opp for host-objektet skifter tilstand. Service-objekter som er definert for et host-objekt som har tilstanden "DOWN" vil automatisk bli vist under "Known service problems" i webgrensesnittet, og ingen sjekker vil bli kjørt på dem før host-en har tilstandeden "UP" igjen.

Grunnsjekken for en host defineres i host-konfigurasjonen med "check_command", som vist under.

```
define host {
...
    check_command check_host_alive
}
```

Variabler

For å kunne gjøre konfigurasjon av Icinga mest mulig dynamisk er det en rekke forhåndsdefinerte variabler en kan bruke i konfigurasjonsfilene. Disse kalles makroer. En fullstendig liste over disse og hvor de kan benyttes finnes i Icinga-dokumentasjonen [27]. "$HOSTADDRESS$" kan for eksempel benyttes i et command-objekt slik at det bare må

skrives en gang, og ikke for hver host. En kan også definere egne variabler som for eksempel "_SSHPORT". Disse kan sendes videre til selve kommandoen som kjøres og benyttes som argumenter til plugin-en som kjører en sjekk, slik:

```
define host {
...
    host_name    example_server
    _SSHPORT     222
}
define command {
    command_name   check_ssh
    command_line   $USER1$/check_ssh -H $HOSTADDRESS$ -p $_HOSTSSHPORT$
}
```

Her benyttes de innebygde variablene "USER1" for banen til plugin-en som kjøres, HOSTADDRESS for IP-en til den hosten som sjekken skal kjøres på og den egendefinerte variabelen _SSHPORT som defineres på host-objektet.

USER-variablene defineres i den globale konfigurasjonsfilen og brukes der hvor informasjon skal sjules fra webgrensesnittet. Som for eksempel når det trengs et passord for å utføre en sjekk.

Templates

En template er ikke et objekt i seg selv, men en mal for et objekt. Med direktivet "register 0" vil ikke Icinga registrere dette objektet for overvåkning. Ved å bruke direktivet "use" kan en i et annet objekt "arve" konfigurasjonen fra template-en. Objektet som er definert som template-er kan også arve, og en kan dermed sette opp et hierarki der en spesifiserer generell konfigurasjon i toppen og arver nedover til mer spesiell konfigurasjon.

I eksempelet under er generell konfigurasjon for alle host-objekter definert i "generic_host", som brukes i de andre objektene. I "generic_firewall" erstattes verdien for kontaktgruppe, og i "cisco_firewall" settes en egendefinert variabel:

Template-en generic_host:

```
define host {
    name                            generic_host    ; The name of this host
        template
    notifications_enabled           1       ; Host notifications are enabled
    event_handler_enabled           1       ; Host event handler is enabled
    flap_detection_enabled          1       ; Flap detection is enabled
    failure_prediction_enabled      1       ; Failure prediction is enabled
    process_perf_data               1       ; Process performance data
    retain_status_information       1       ; Keep status after Icinga restart
    retain_nonstatus_information    1       ; Keep non-status after Icinga
        restart
    check_command                   check-host-alive
    max_check_attempts              10
    normal_check_interval           5
    retry_check_interval            2
    notification_interval           0
    notification_period             24x7
    notification_options            d,u,r
```

```
    contact_groups                  all_contacts
    register                        0       ; Not a real host, just a template
}
```

Template-en generic_firewall, som arver fra generic_host:

```
define host {
   name             generic_firewall
   use              generic_host
   contact_groups   firewall_admins
   register         0
}
```

Template-en cisco_firewall, som arver fra generic_firewall:

```
define host {
    name            cisco_firewall
    use             generic_firewall
    register        0
    _WANPORT        WAN ; Custom variable for WAN port to monitor
}
```

Host-objektet hig-hw1, som arver fra cisco_firewall:

```
define host {
    use             cisco_firewall
    host_name       hig-fw1
}
```

Dette gjør at det blir lite konfigurasjon hver gang en ny brannmur skal legges til.

Regulære uttrykk

Ved å sette "use regular expressions" i konfigurasjonen for Icinga, kan en benytte regulære uttrykk i attributter i et objekt, der en refererer til andre objekter. Icinga støtter standarden POSIX Extended Regular Expressions, og vil automatisk tolke navnet på objekter som et regulært uttrykk dersom det inneholder `"*", "?", "+", eller "\"`.

Dette kan være nyttig dersom en for eksempel vil legge alle hoster i en hostgroup, som har en navnekonvensjon som gjør at hver av hostene unikt kan identifiseres ved å bytte ut deler av navnet med en variabel.

I eksempelet nedenfor brukes det regulære uttrykket `"^HiG\[0-9\]+\$"` i attributten "members" til å legge til alle konfigurerte hosts der navnet starter på "HiG" etterfulgt og avsluttet med ett eller flere siffer mellom 0 og 9, i hostgroup-objektet "all_servers".

```
define hostgroup {
    hostgroup_name  all_servers
    alias           All Servers
    members         ^HiG[0-9]+$
}
```

Det kan også benyttes til å ekskludere en host fra en service-sjekk. I eksempelet nedenfor er HiG3 medlem av hostgroup-en "windows_servers", men blir ekskludert fra å kjøre sjekken på diskplass ved å sette "!HiG3":

```
define service {
    use                     generic_service
    service_description     Disk space
    hostgroup_name          windows_servers
    host_name               !HiG3
    check_command           win_nrpe!CheckDriveSize!Drive="C" MaxWarnUsed=80%
        MaxCritUsed=90%
}
```

2.9 Avhengigheter

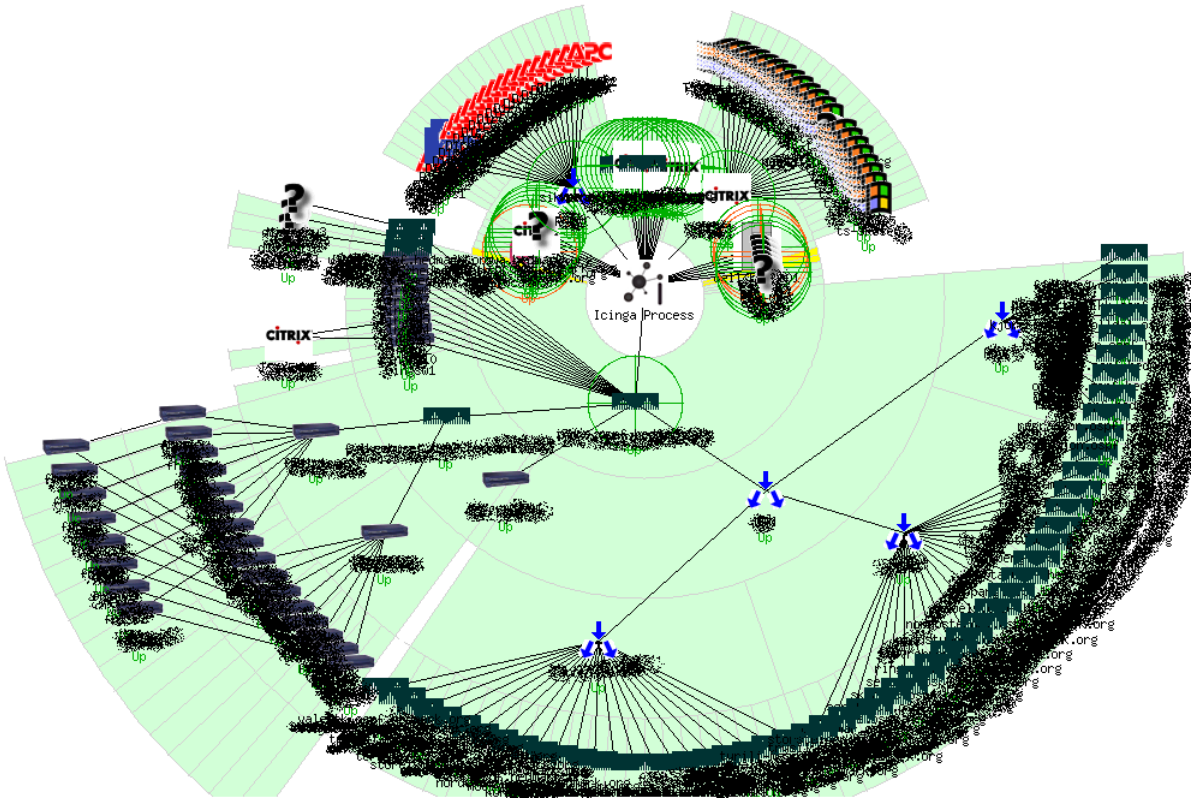
I Icinga kan en definere tre ulike typer avhengigheter, "Parent", "Service dependency", og "Host dependency". Disse har innvirkning på hvilke varsler som sendes ut og statusen de ulike service- og host-objektene får. Hver av disse krever egen konfigurasjon.

2.10 Parent

Et host-objekt kan settes som en parent til et annet host-objekt. Host-objekter som har en parent blir betegnet som en child-host eller child-service. Dette brukes primært for å unngå varsler om andre host- og service-objekter enn det som er definert som parent, om denne blir utilgjengelig. For eksempel hvis kjernerouteren mister strømtilgangen, ønskes ikke varsler for eventuelle host-objekter og tilhørende service-objekter som har kjernerouteren som parent. Disse vil ikke lenger bli sjekket og tilstanden vil bli satt til "UNREACHABLE". Tjenester som tilhører host-ene vil få tilstanden "UNKNOWN".

For redundante oppsett kan et host-objekt ha flere "parents" definert. Host-objektet vil da ha tilstanden UP så lenge minst én "parent"-host er UP.

Ut ifra parent-relasjonene mellom host-objekter i konfigurasjonsfilene genereres også et "Status Map" over nettverket, som vist i Figur 2.6. Dette brukes til å visualisere alle relasjonene, og vil gjengi redundante oppsett og hvilke host-objekter som er påvirket av at parent-hosten er DOWN.



Figur 2.6: Kart som viser parent- og host-relasjoner

2.11 Service dependency

Service dependency er en egen objekttype der en kan sette opp avhengighet mellom to service-objekter. Formålet med dette er å unngå varsel hvis en tjeneste får tilstanden DOWN, som følge av at en annen tjeneste har det. For eksempel bør det ikke varsles om at tjenester som er avhengig av autentisering via LDAP-serveren får "Access Denied"-feilmeldinger, hvis LDAP-serveren er DOWN. Det service-objektet andre service-objekter er avhengige av kalles en master service.

Icinga undersøker om avhengigheter er definert for et service-objekt, før det utføres sjekker på det. Dette er med på å bestemme om det blir sendt ut varsel for service-objektet.

I eksempelet under er tjenesten “Check SMB” på HiG3 avhengig av master-service-en Check LDAP på HiG3. “execution_failure_criteria” bestemmer ved hvilke tilfeller tjenesten ikke skal sjekkes. Her er den satt til “c”, som vil si at Check SMB ikke skal kjøres dersom “Check LDAP” er i CRITICAL tilstand.

```
define servicedependency {
    host_name                       HiG3
    service_description             Check LDAP
    dependent_host_name             HiG3
    dependent_service_description   Check SMB
    execution_failure_criteria      c
    notification_failure_criteria   c
}
```

Ved standard konfigurasjon vil Icinga benytte siste harde tilstand for master service. Dersom “max_check_attempts” for eksempel er satt til 4, vil ikke Check SMB stoppes fra å kjøres før Check LDAP har blitt sjekket fire ganger, og oppnår en hard tilstand.

På attributten notification_failure_criteria kan det settes hvilke service-tilstander en ikke skal varsle for. Dette er hovedgrunnen til at det settes opp slike avhengigheter. Når denne er satt til “c” vil det ikke varsles om master service-en får tilstanded CRITICAL.

Service-objekter har kan også arve avhengighetene til en master service ved å sette direktivet “inherits_parent 1”. For eksempelet over vil Check SMB da arve fra service-objekter som Check LDAP arver fra.

For å konfigurere avhengigheter mellom tjenester som kjører på samme host kan en utelate “dependent_host_name” I eksempelet under vil alle tjenester på host-objektet være avhengig av tjenesten “NRPE-daemon”.

```
define servicedependency{
    ;dependent_host_name        ; Not defined to make dependancy on same host
    hostgroup_name               linux_servers
    service_description            NRPE-daemon
    dependent_service_description  NRPE Check *
    execution_failure_criteria     c
    notification_failure_criteria  c
}
```

En kan også erstatte host_name og dependant_host_name med hostgroup og dependant_host_group for å lage avhengigheter på alle objekter som tilhører gruppen.

2.12 Host dependency

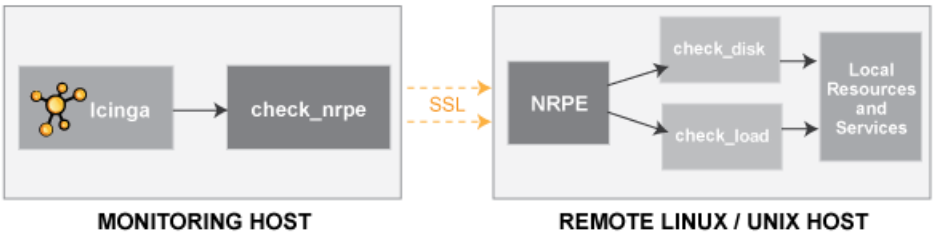
For å sette en avhengighet mellom to host-objekter benyttes host dependency. Dette må ikke forveksles med parent, som refererer til nettverksoppsettet. En host dependency vil si at et host-objekt er avhengig av et annet host-objekt. Host dependency er bare nyttig i veldig spesielle tilfeller, da det for de fleste tilfeller vil være avhengigheter til en eller flere tjenester på host-en [28], eller at en host er avhengig av å kjøre en sjekk igjennom en annen host.

2.13 Protokoller og agenter

For å kjøre sjekker på enheter må Icinga-serveren kunne kommunisere med operativsystemene på de ulike enhetene. Icinga henviser til protokollene SSH, SNMP, RPC (WMI) og agentene NRPE og NSCA (en del av pakken NSClient++) i dokumentasjonen [29,30].

NRPE

NRPE (Nagios Remote Plugin Executor) består av plugin-en "check_nrpe" på Icinga-serveren og en daemon som installeres på hver server som skal overvåkes. NRPE eksekverer lokale plugin-er på den eksterne serveren og returner dataen fra denne til check_nrpe. Tillatelse for eksekveringer defineres i en konfigurasjonsfil på hver server, der en oppgir hvilke IP-adresser som kan koble seg til og hvilke kommandoer som er gyldige.



Figur 2.7: NRPE

SSH

Gjennom en tilkobling til en SSH (Secure Shell)-server kan en spesifisere en kommando som skal kjøres på en ekstern maskin, og hente output-en til det opprinnelige shellet. Dette virket som den beste måten å overvåke Linux-servere på, ved starten av prosjektet. SSH-prosjektet er meget utbredt og blir aktivt sjekket for sikkerhetshull. SSH er også installert på de aller fleste Linux-servere, og en vil dermed unngå å lytte på en ekstra port.

Ulempen med å kjøre sjekker over SSH er at en må sette opp nøkkelbasert innlogging mellom Icinga-serveren og alle Linux-serverne som skal overvåkes. En kan begrense rettigheter for brukeren som kjører sjekkene, men den vil fortsatt ha muligheten til å logge inn til et shell og eksekvere kommandoer. Dette kan føre til at en potensiell angriper via Icinga-serveren, har en vei inn til alle servere som overvåkes, ved å bruke den private nøkkelen.

SSH gir også endel overhead både på nettverkstrafikk og i CPU-tid [31]. En måte å begrense dette på er å benytte SSH med ControlMaster. Dette innebærer en endring i SSH-konfigurasjonen på serveren, slik at SSH-klienten lagrer informasjon om hver utgående TCP-tilkobling, og samme tilkobling kan benyttes mot samme host. Dermed unngås en ny TCP-tilkobling for hver ny sjekk som skal utføres på hosten.

SNMP

Det meste av nettverksutstyr beregnet for bedrifter, har i dag støtte for protokollen SNMP (Simple Network Management Protocol [32]). SNMP definerer en enkel og effektiv måte å overvåke enheter på og det gir en standard som leverandørene følger.

SNMP baserer seg på variabler som blir gjort tilgjengelig på enheten som skal overvåkes (kalt agenten). Disse variablene er definert i en Management Information Base (MIB). MIB-filene er ofte tilgjengelige på produsentens hjemmeside. Her beskrives strukturen for dataen i et hierarkisk navnerom som inneholder Object Identifier (OID). Hver OID identifiserer en variabel som kan bli lest eller satt via SNMP. Enheten som ber om disse variablene kalles "manager".

En ulempe med SNMP er at OID-ene er lange og tunge å lese. For eksempel for å hente ut batterikapasiteten på en APC UPS benyttes OID-en:

```
1.3.6.1.4.1.318.1.1.1.2.2.1.0
```

Dette kan oversettes til:

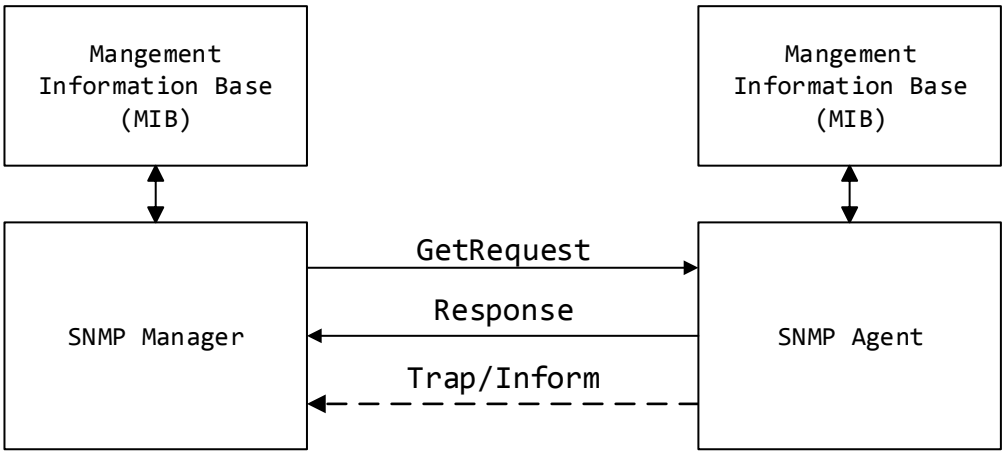
```
iso(1). org(3). dod(6). internet(1). private(4). enterprises(1). apc(318).
    products(1). hardware(1). ups(1). upsBattery(2). upsAdvBattery(2)
    upsAdvBatteryCapacity.0
```

Denne OID-en er definert i APCs MIB-fil "PowerNet MIB". Ved å laste inn denne kan en også benytte "PowerNet-MIB::upsAdvBatteryCapacity.0".

SNMP finnes i flere versjoner. I dag brukes for det meste versjon 2c og den nyeste, versjon 3. De ulike versjonene er ikke kompatible, men i praksis støtter det meste av utstyr som støtter v3 eller v2c også v1 [33]. Hovedforskjellen på versjon 2c og 3 er at versjon 3 har støtte for flere sikkerhetsmekanismer, men krever noe mer konfigurasjon. Sikkerhetsaspektet ved dette er diskutert videre i Kapittel 4.

Det eksisterer to muligheter når enheter skal overvåkes via SNMP. Enten kan enhetene selv rapportere inn hendelelser via SNMP trap/inform, eller så kan serveren spørre enhetene via SNMP GetRequest. Valg av det første medfører at en kan gjøre varsling umiddelbart, men da må også hver enhet konfigureres med parametere for hvilke trap-meldinger som skal sendes og til hvilken IP-adresse. Icinga har ingen støtte for å motta trap-er, men en kan installere en egen daemon for dette og rapportere inn data med passive sjekker.

For å benytte SNMP trap/inform må en tillate tilkoblinger til overvåkingsserveren i stedet for bare fra denne. Hvilke trap-meldinger som skal sendes må også konfigureres på hver enhet, og en vil da ikke kunne ha alle konfigurasjoner sentralisert på Icinga-serveren.



Figur 2.8: Overvåkning med SNMP

SNMP benytter seg av UDP for å levere trap-meldinger. UDP er en såkalt upålitelig protokoll, noe som vil si at det ikke er noen omsending av pakker eller noen tilbakemelding (ack) for at pakkene kommer fram. Derfor kan en risikere å miste en trap. SNMP inform, som kom i SNMPv3, løser dette ved at agenten vil sende en ny inform-pakke i et intervall, helt til manageren gir beskjed om at trap-meldingen er mottatt.

For å benytte SNMP på Windows- og Linux-servere kreves installasjon av en SNMP-agent [34,35].

WMI (Windows Management Instrumentation)

WMI er et grensesnitt som gjør det mulig å lage programmer eller script som utfører oppgaver mot operativsystemet Windows, både lokalt eller på eksterne maskiner. Disse oppgavene kan være alt fra å restarte maskinen, til å hente ut logger over hendelser som har inntruffet på maskinen. I en overvåkningsløsning vil det være mest relevant å kjøre kommandoer som henter ut informasjon om maskinen og tjenester som kjører på den. Dette kan for eksempel være hvilke prosesser som bruker mest CPU-tid.

For å benytte WMI-spørringer direkte fra en ekstern enhet må det konfigureres brukertilgang og brannmurregler [36]. Med NSclient++ har en mulighet til å kjøre WMI-spørringer over NRPE uten ekstra konfigurasjon.

Valg av Agenter

Valg av protokoll eller agenter har basert seg på følgende punkter:

- Ressursbruk (CPU)
- Nettverkstrafikk
- Utrulling
- Sikkerhet

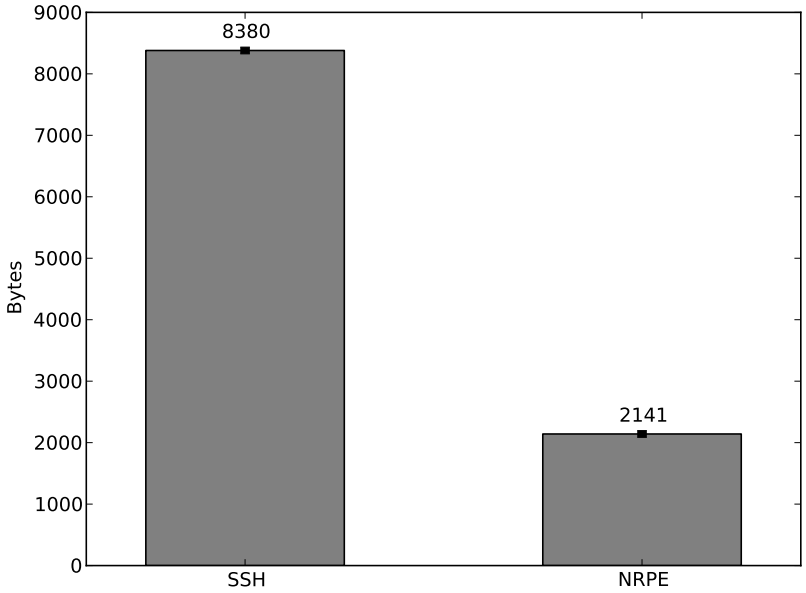
Nettverkstrafikken som ble generert ved både en NRPE- og en SSH-sjekk ble målt ved å benytte et Bash-script som kjørte en disk-sjekk 100 ganger, som vist under.

```
#!/usr/bin/env bash
if [ "$1" == "ssh" ]; then
    cmd="/usr/lib/nagios/plugins/check_by_ssh -H 10.60.0.21 -C \"/usr/lib/nagios/
        plugins/check_disk -W 10% -C 5% -M -A\" > /dev/null"
elif [ "$1" == "nrpe" ]; then
    cmd="/usr/lib/nagios/plugins/check_nrpe -H 10.60.0.21 -c check_all_mounts -a
        10,5 > /dev/null"
else
    echo "You must specify ssh or nrpe as the first argument"
    exit 1
fi
for i in {1..100}; do
    eval $cmd
done
exit 0
```

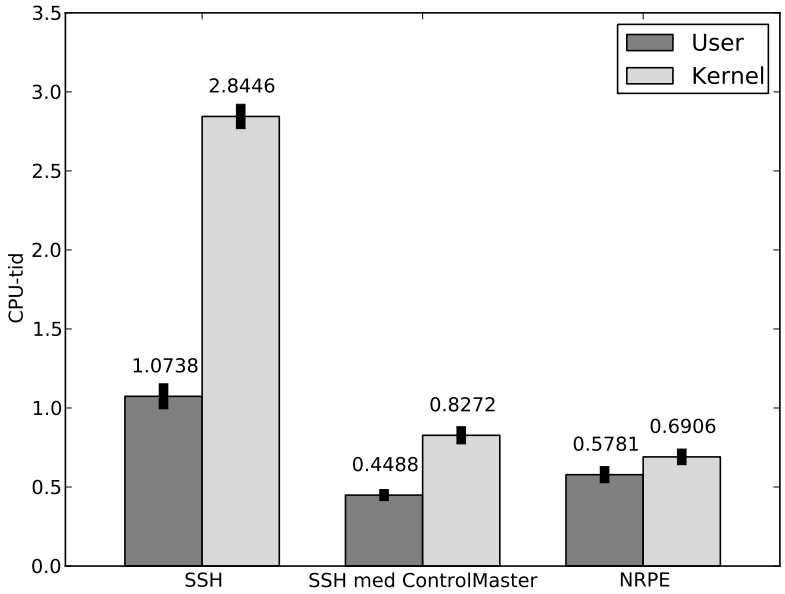
Både serveren og klienten stod i et eget nettverk i Virtualbox sammen med en Windows-maskin med et nettverkskort satt i promiscious mode, for å kunne sniffe nettverkstrafikken. Denne fanget nettverkstrafikken med Wireshark. Trafikken mellom de to maskinene ble hentet ut og data for hver tilkobling ble brukt til å kalkulere båndbreddebruk.

Det ble også undersøkt hvor mye CPU-tid NRPE og SSH trenger for å utføre sjekker. Fordi hver sjekk går veldig raskt ble det kjørt 100 disksjekker, 100 ganger. "time"-kommandoen ble brukt for å se på tiden brukt i user- og kernel-mode. Dette ble gjort ved å bruke Bash-scriptet nedenfor, som kjører det samme scriptet som ble brukt for å generere nettverkstrafikk.

```
#!/usr/bin/env bash

args=(nrpe ssh)

for arg in "${args[@]}"; do
        for i in {1..100}; do
                /usr/bin/time -f "%U\t%S" run_check.sh $arg
        done
done
exit 0
```



Figur 2.9: Nettverkstrafikk



Figur 2.10: CPU-bruk

For en utrulling i den skalen som vil være aktuelt for IKT-avdelingen vil ikke denne nettverkstrafikken være noen stor faktor. Figur 2.9 viser at gjennomsnittlig vil total båndbreddebruk for en sjekk var 8380 byte. Selv med 1000 enheter som overvåkes over SSH, der hver kjører 100 sjekker hvert 5. minutt, vil den totale båndbreddebruken, dersom alle sjekkene var jevnt fordelt over 5 minutters-intervallet:

$$\frac{8.38\,kB \times (1000\,(enheter) \times 100\,(sjekker)) \times 8\,b/B}{5\,(min) \times 60\,s} = 22346.67\,Kb/s = 22.35\,Mb/s$$

	CPU-tid i user(s)	**CPU-tid i kernel(s)**	**Nettverkstrafikk (B)**
SSH	1.0738 (0.0792)	2.8446 (0.0771)	8380 (101.38)
NRPE	0.5781 (0.0347)	0.6906 (0.0495)	2141 (55.08)
SSH med ControlMaster	0.4488 (0.0509)	0.8272 (0.0543)	1700*

*For SSH med ControlMaster var det ikke mulig å separere ut hvor mye trafikk det var på én sjekk. Her er det brukt et gjennomsnitt ved å dele den totale mengden trafikk på antall sjekker.

Tabell 2.3: Test av CPU-bruk ved NRPE og SSH

På et gigabit-nettverk, som i tillegg er full duplex, vil ikke dette være nevneverdig. Icinga vil forsøke å fordele sjekkene utover slik at lasten på Icinga-serveren og enhenetene som overvåkes blir minst mulig [37], men i praksis blir ikke dette helt jevnt fordelt, tallet kan derfor bli noe høyere.

Resultatene av testen av CPU-tid i 2.10, viser at SSH krever mer CPU-tid enn NRPE. Mye av dette er på grunn av overhead ved tilkobling og frakobling, som tallene for SSH med ControlMaster viser.

Det ble valgt å benytte NRPE både for Linux- og Windows-servere. En fordel med dette er at samme protokoll for alle tilkoblinger fra Icinga-serveren til samme port (5666) på andre servere kan benyttes. Dette gjør at det kun trengs én brannmurregel for Icinga, som igjen holder antall angrepsvektorer nede. En annen fordel er at samme service- og command-objekter kan benyttes i Icinga for sjekker som skal utføres via NRPE, uavhengig av om hosten kjører Windows eller Linux.

Kapittel 3

Implementasjon

Dette kapittelet handler om hvordan gruppen har utført implementasjonen av overvåkningsløsningen for IKT-avdelingen, hvilke valg som er tatt underveis og hvordan de forskjellige enhetene overvåkes.

3.1 Utstyr

Labmiljø

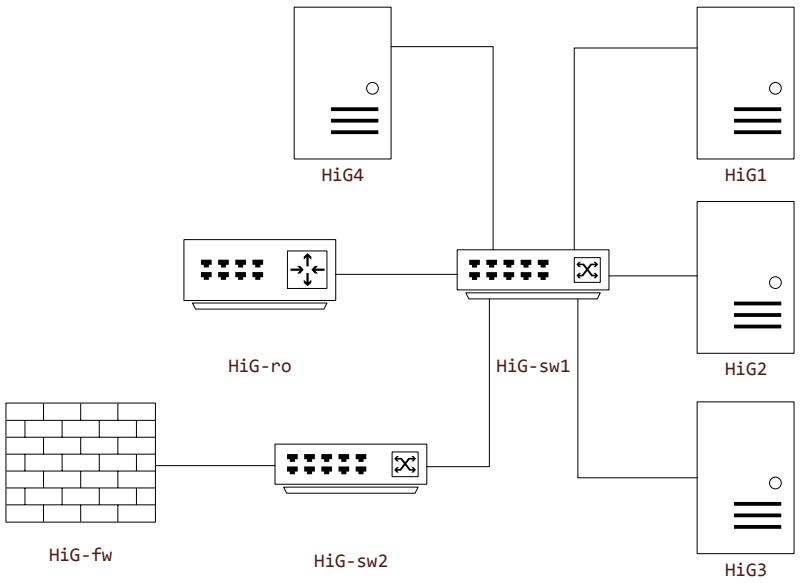
Et labmiljø har vært brukt for å teste plugin-er og script før de ble implementert på produksjonsserveren. Ved å teste i et labmiljø først, kan en se hvordan sjekker oppfører seg, før de i stor skala implementeres i produksjon. Labmiljøet inneholder utstyr og tjenester som gjenspeiler det IKT-avdelingen benytter. På denne måten kan ulike scenarier og utstyr testes før dette settes i produksjon. I Tabell 3.1 er det en oversikt over utstyret i labmiljøet.

Type	**Beskrivelse**	**Dato installert**	**Tjenester**
Server	Debian linux (HiG1)	22.01.2013	Icinga, Icinga-Web, Icinga-mobile, MySQL, Apache
Server	Debian linux (HiG2)	22.01.2013	MySQL, Apache
Server	Windows 2008 R2 (HiG3)	22.01.2013	DNS, DHCP, AD, IIS, Fileserver, MSSQL
Server	Windows 2008 R2 (HiG4)	19.02.2013	Exchange
Switch	Cisco 3550 (HiG-sw1)	29.01.2013	SNMP
Switch	Dell Powerconnect 5324 (HiG-sw2)	29.01.2013	SNMP
Router	Cisco 2600 (HiG-ro)	05.02.2013	SNMP
Firewall	Cisco 515E (HiG-fw)	05.02.2013	SNMP

Tabell 3.1: Labmiljø

Serverne er virtuelle maskiner i et eget VLAN, som er tilgjengelig på fysiske porter, slik at nettverksutstyret kan plasseres i samme nettverk. VLAN-et har også tilgang ut mot Internett og har vært tilgjengelig for gruppen over VPN. Tjenester som testes på HiG1, HiG2, HiG3 og HiG4 blir alle overvåket via NRPE. For nettverksutstyret blir SNMP benyttet.

I Figur 3.1 vises det logiske oppsettet av labmiljøet. I Tabell 3.1 vises hvilke tjenester som kjøres. HiG-fw, HiG-sw1 og HiG-sw2 er koblet i serie for å teste avhengigheter og følgefeil.



Figur 3.1: Labmiljø

Produksjonsserveren

Spesifikasjonene på bladeserveren:

- 4 CPU-er med 4 kjerner à 2.4 GHz
- 32 GB RAM

Programvare:

- Debian 6
- Apache2
- MySQL
- Icinga 1.8.4
- SNMPtrapd
- SNMPtt
- Graphite
- Metricinga
- sendmail

Enhet for overvåkning av servermiljø

For å overvåke temperatur og luftfuktighet på serverrommet ble det i samråd med IKT-avdelingen kjøpt inn en APC NetBotz 200, som støtter opptil 12 eksterne sensorer [38]. Denne oppfyller kravene gitt i oppgavebeskrivelsen.

3.2 Produksjonsserver

"Quis custodiet ipsos custodes?" er et latinsk uttrykk som kan oversettes med "hvem passer på de som passer på?". I en overvåkningsløsning er det viktig å stille spørsmålet; hva skjer hvis overvåkningsserveren går ned? Et forslag ved starten av prosjektet var å legge overvåkningsserveren på et Xen- eller VMware-cluster. Men dette ble etter noe omtanke stemplet som en dårlig idé. Dersom clusteret gikk ned, ville også overvåkningsserveren gå ned. Det ble derfor bestemt at Icinga-serveren skulle være en egen fysisk server, med databasen installert lokalt.

For å sikre tilgjengeligheten til Icinga ytterligere, er det også mulig å sette opp et redundant oppsett der alle Icinga-installasjoner kan dele resultater av sjekker mellom seg. Ekstra viktig vil et slikt oppsett være dersom en knytter overvåkningssystemet mot SLA-er. Dersom en mister data om oppetid og tilgjenglighet på en tjeneste, vil en ikke lenger kunne vise hva den har vært.

En annen utfordring var å vite hvor kraftig hardware serveren trengte. Her ble referanselisten til Icinga lagt til grunn, hvor mange organisasjoner har lagt inn informasjon om sine oppsett [39]. Etter avtale med oppdragsgiver ble det bestemt å sette opp en bladeserver, som kan oppgraderes dersom det skulle bli nødvendig.

Debian 6 ble valgt som operativsystem på Icinga-serveren etter ønske fra oppdragsgiver.

3.2.1 Installasjon

Ved starten av iterasjonen "kjerneprogramvare" var den nyeste versjonen av Icinga 1.8.4. I pakkebrønnen for debian-stable fantes bare versjon 1.0.2 av Icinga, i backports lå 1.7.1. Icinga opprettholder en egen pakkebrønn - "The Debian Monitoring Project" (debmon [40]). Fra denne kunne versjon 1.8.4 installeres. I samråd med teknisk kontakt ved IKT-avdelingen ble det bestemt å bruke versjon 1.8.4 fra debmon.

Icinga Webgrensesnitt

Webgrensesnittene Icinga Classic og Icinga Web ble installert med pakkene "icinga-classic" og "icinga-web 1.8.1", via pakkebrønnen debmon.

Tidlig ble det avgjort at Icinga Web primært skulle brukes som webgrensesnitt mot Icinga. Denne avgjørelsen ble tatt fordi Icinga Web kommer med et REST-API, har støtte for LDAP-integrasjon og aksesstyring på mange objekter.

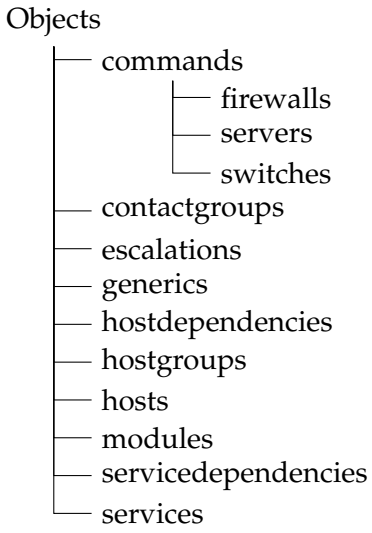
Etter testing ble det oppdaget at Icinga Web opplevdes som mye tregere enn Icinga Classic og inneholder en feil som ble rapportert tilbake til Icinga Development Team. Feilen som ble rapportert inn omfatter søk på hostgroup-objekter eller servicegroup-objekter (bugreport [41]). Det ble derfor lagt opp til at både Icinga Classic og Icinga Web kan benyttes på lik linje.

3.2.2 Konfigurasjonsfiler

Ved standard installasjon av Icinga er konfigurasjonen delt opp i objekttyper med flere objekter i hver fil:

- contacts_icinga.cfg
- generic-host_icinga.cfg
- hostgroups_icinga.cfg
- localhost_icinga.cfg
- timeperiods_icinga.cfg
- extinfo_icinga.cfg
- generic-service_icinga.cfg
- services_icinga.cfg
- commands.cfg

Dette var uoversiktlig og en mer oppdelt konfigurasjon var ønskelig, som også er anbefalt ved større installasjoner [42, 43]. Derfor ble det bestemt å sette opp følgende hovedinndeling (med undermapper videre der det var hensiktsmessig):



Det ble testet ut et par verktøy for å administrere konfigurasjonsfilene, NConf [44] og NagiosQL [45] i et webgrensesnitt. Disse ble valgt bort til fordel for manuell konfigurering, da oppdeling av konfigurasjonen ikke var støttet, og de kunne ikke kombineres med manuell konfigurering. I samråd med oppdragsgiver ble det avgjort at manuell konfigurasjon oppfyller kravet om at det skal være enkelt å legge til nye enheter for overvåking.

3.2.3 Bruk av hostgroup

Som nevnt i 2.6 knyttes et service-objekt til et host-objekt og et command-objekt, for at det skal kjøres en sjekk. For å slippe å skrive et service-objekt for hvert nytt host-objekt, benyttes gruppering av host-objekter i en hostgroup. Under vises et eksempel på hvordan dette er satt opp for MySQL-servere.

Først defineres en hostgroup for alle SQL serverne. Dette gjøres for å gruppere alle SQL serverne uavhengig av hvilken databasetype som brukes.

```
define hostgroup {
    hostgroup_name      sql_servers
    alias               SQL Servers
    hostgroup_members   mysql_servers, mssql_servers, oracle_servers
}
```

Deretter defineres en hostgroup for alle MySQL-servere.

```
define hostgroup {
    hostgroup_name      mysql_servers
    alias               MySQL Servers
}
```

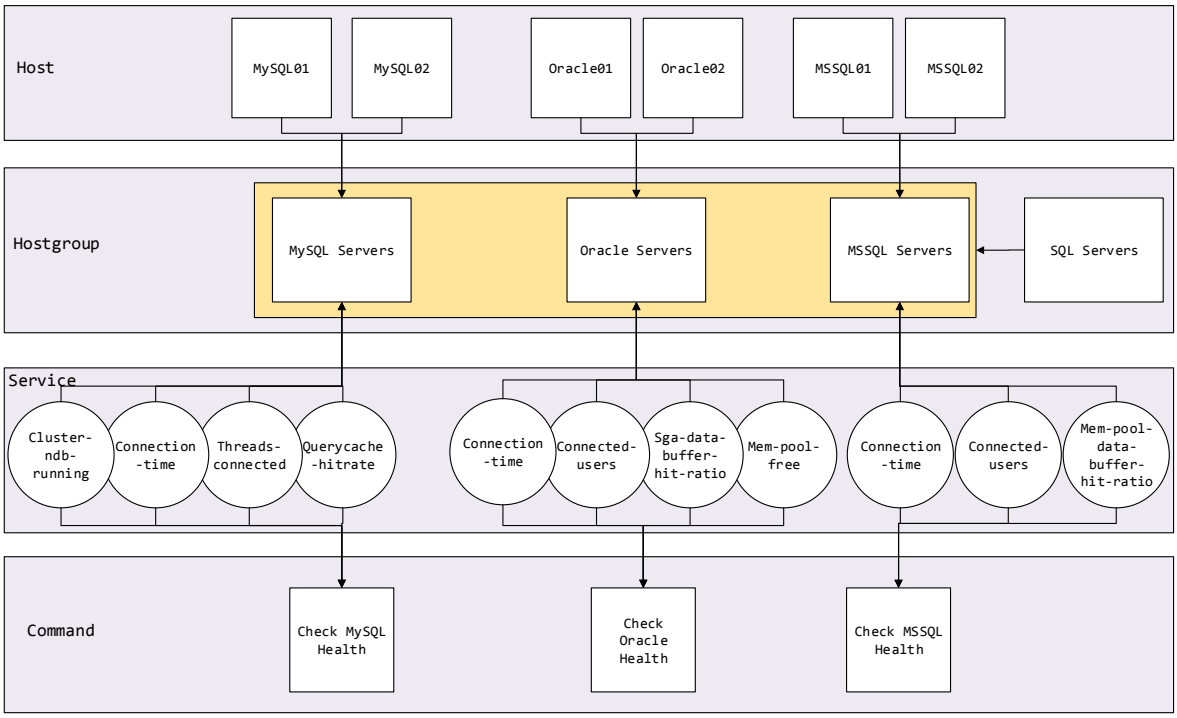
For å legge til en host i denne gruppen kan host-objektet være definert på følgende måte.

```
define host {
    use             generic_linux_host
    address         10.60.0.21
    host_name       HiG2
    alias           HiG2
    hostgroups      debian_servers, mysql_servers
}
```

Service-objektet defineres slik og refererer til hostgroup-en og kommandoen "check_mysql_health". Alle host-objekter i "mysql_servers" vil da få denne sjekken.

```
define service {
    service_description MySQL Connection Time
    use                 generic_service
    name                mysql_connection_time
    hostgroup_name      mysql_servers
    check_command       check_mysql_health!connection-time!0.1!0.4
}
```

Figur 3.2 viser en visuell fremstilling av hvordan dette henger sammen for alle SQL-servere:



Figur 3.2: Visualisering av konfigurasjon av SQL-servere

3.3 Generering av grafer

I utgangspunktet ble det bestemt at ytelsesdata skulle holdes utenfor oppgaven. Det ble likevel satt opp en løsning for å visualisere ytelsesdata fra sjekkene med programmet Graphite [46]. Dette ble gjort fordi det var ønskelig å kunne etablere en baseline for tjenestene slik at bedre grenseverdier kunne settes.

For å sette opp eksportering av ytelsesdataen benyttes et vanlig command-objekt i Icinga:

```
define command {
    command_name    rotate_perf_service
    command_line    /bin/mv /usr/local/icinga/var/perfdata/service-perfdata /usr
        /local/icinga/var/perfdata/logs/service-perfdata.$TIMET$
}
```

Icinga.cfg må konfigureres til å benytte dette:

```
process_performance_data=1
service_perfdata_file=/usr/local/icinga/var/perfdata/service-perfdata
service_perfdata_file_processing_command=rotate_perf_service
service_perfdata_file_template=[SERVICEPERFDATA]\tDATATYPE::SERVICEPERFDATA\
    tTIMET::$TIMET$\tHOSTNAME::$HOSTNAME$\tSERVICEDESC::$SERVICEDESC$\
    tSERVICEPERFDATA::
    $SERVICEPERFDATA$service_perfdata_file_processing_interval=200
```

For å transformere ytelsesdataene til riktig format for Graphite benyttes "Metricinga" [47]. Dette scriptet sjekker "spool"-mappen som command-objektet flytter filene med ytelsesdata til. Metricinga sjekker mappen én gang i minuttet etter filer som enda ikke er prosessert, og sender data inn til Graphite (via carbon). Gjennom Graphite vil det dermed genereres grafer basert på ytelsesdata fra alle service-sjekker som gir dette.

Det ble også testet å modifisere scriptet til å lagre outputen til service-sjekkene direkte i en MySQL-database, som vist i Vedlegg H.7. Dette ble gjort for å oppfylle kravet fra oppgavebeskrivelsen om at alle henvendelser skal lagres i database. I samråd med oppdragsgiver ble det avgjort å gå bort i fra dette fordi det resulterte i et høyt antall nye rader over kort tid. Som et alternativ til dette kan en ta inn all data til Graphite, men aggregere det etter en viss tid. For eksempel lagre dagsgjennomsnittet for data som er eldre enn seks måneder. Dataene kan da eksporteres fra Graphite til videre bruk.

Metricinga kommer ikke med init-script for Debian. For at prosessen skal kunne startes automatisk med serveren ble dette skrevet av gruppen (Vedlegg H.5).

3.4 Overvåkning av Windows-servere

For overvåkning av Windows-servere brukes programmet NSClient++ [48]. Programmet brukes for å kommunisere med ulike agenter over ulike protokoller på en ekstern server. I dette prosjektet brukes NSClient++ til å hente ut informasjon via NRPE-agenten og å kjøre WMI-spørringer mot en Windows-server.

Under installasjon av NSClient++ blir en standard konfigurasjonsfil generert. Her derfineres hvilke IP-adresser som kan koble seg til, og hvilke sjekker som er tilgjengelige.

NSClient++ er valgt fordi klienten oppdateres hyppig [49], og det er den agenten som blir referert i Icinga/Nagios dokumentasjon [50]. Med NSClient++ kommer også forhåndskonfigurerte plugin-er, for eksempel for å sjekke CPU, minne og harddiskplass.

3.5 Overvåkning av Linux-servere

Overvåkning av Linux-servere skjer utelukkende ved bruk av NRPE. For Debian-servere kan en NRPE-daemon installers fra pakkebrønnen "stable" med kommandoen:

```
apt-get install nagios-nrpe-server nagios-plugins-basic
```

For Red Hat og CentOS må det installeres en tredjeparts pakkebrønn som "EPEL" eller "DAG" før nrpe-server kan installeres med yum.

Ved bruk av DAG må "rpm"-pakken "rpmforge" installeres for å kunne bruke pakkebrønnen, slik:

```
rpm -Uvh http://apt.sw.be/redhat/el6/en/x86_64/rpmforge/RPMS/rpmforge-release
    -0.5.2-2.el6.rf.x86_64.rpm
```

For EPEL må pakken "epel" installeres:

```
sudo rpm -Uvh http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.
    noarch.rpm
```

En kan verifisere at pakkebrønnene ble lagt til ved å kjøre en listing av direktivet /etc/yum.repos.d/:

```
$ ls -1 /etc/yum.repos.d/epel*
/etc/yum.repos.d/epel.repo
/etc/yum.repos.d/epel-testing.repo
```

Installasjon av NRPE via yum etter at pakkebrønnene er lagt til:

```
sudo yum install nagios-nrpe-2.12-1.el6.rf.x86_64.rpm nagios-plugins
```

Eksemplene er for CentOS og RHEL 6.x 64-bit.

Konfigurasjonen på Linux-servere skjer på samme måte som på Windows-servere. Det følger ikke med noen plugin-er når en installerer nagios-nrpe-server, derfor installeres også den offisielle plugin-pakken for Nagios. I debian heter denne pakken "nagios-plugins-basic", og for Red Hat/CentOS "nagios-plugins".

3.6 Utrulling av agenter

NSClient++ kan lastes ned som en MSI-pakke, som kan distribueres til servere med en GPO. Konfigurasjonsfilen må enten legges inn i MSI-pakken eller pushes over GPO for seg selv. Det ble laget en veiledning på hvordan denne klienten skal installeres, og hvilke opsjoner som er relevante. Denne finnes i Vedlegg D.

For Linux-servere installeres pakkene via pakkebehandleren, som nevnt i 3.5. For utrulling kan et script som kobler seg til og kjører kommandoen for installering skrives. Konfigurasjonsfilen kan pushes over SCP.

Distribuering er ikke benyttet fordi IKT-avdelingen ønsker å gjøre installasjonen manuelt for å ha mest mulig kontroll, sikre seg mot uforutsette problemer og gjøre utrulling i faser.

For annen infrastruktur brukes for det meste SNMP for å hente ut informasjon. Dette konfigureres på hver enkelt enhet, og krever ikke noe ekstra programvare installert. I noen tilfeller brukes egne API-er for å hente ut informasjon, som for eksempel for VMware. Her må Icinga serveren ha tilgang til å bruke API-et, som konfigureres på VirtualCenter-serveren.

3.7 Lokale ressurser

For både Linux- og Windows-servere er det satt opp noen grunnsjekker som skal kjøres på alle servere. Dette er CPU-last, harddiskplass og minnebruk. Hver av sjekkene er definert i et service-objekt der hostgroup-ene er satt til “windows_servers” og “linux_servers”.

For alle disse sjekkene er det mulig å angi grenseverdier både som prosentandel og absolutte tall. Det kan også sjekkes mot både andel ledig og andel brukt.

3.7.1 CPU

Overvåkning av CPU vil kunne hjelpe til med å oppdage ressursproblemer. Dette kan komme av flere CPU-krevende applikasjoner på samme server, eller at en applikasjon bruker all CPU-kraft.

På Windows-servere benyttes “CheckCPU” fra NSClient++ for å sjekke CPU-last over NRPE. Her legges det ved tre opsjoner i sjekken som spesifiserer tidsintervallet for datagrunnlaget, grenseverdi for når det skal gis en WARNING og grenseverdi for CRITICAL.

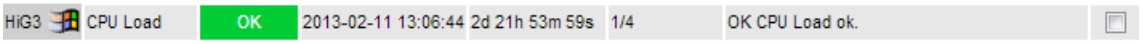
Kommandoen for plugin-en slik den er definert i command-objektet. Dette henter et gjennomsnittsbruk av CPU over 5 minutter:

```
./check_nrpe -u -H 10.60.0.22 -p 5666 -c CheckCPU -a time=5m warn=80 crit=90
```

Svar:

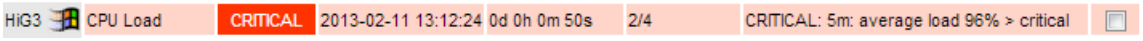
```
OK CPU Load ok.|'5m'=0%;80;90
```

I Figur 3.3 vises svaret fra CPU-sjekken i Icinga.



Figur 3.3: Visning av en CPU-sjekk som gir OK i Icinga

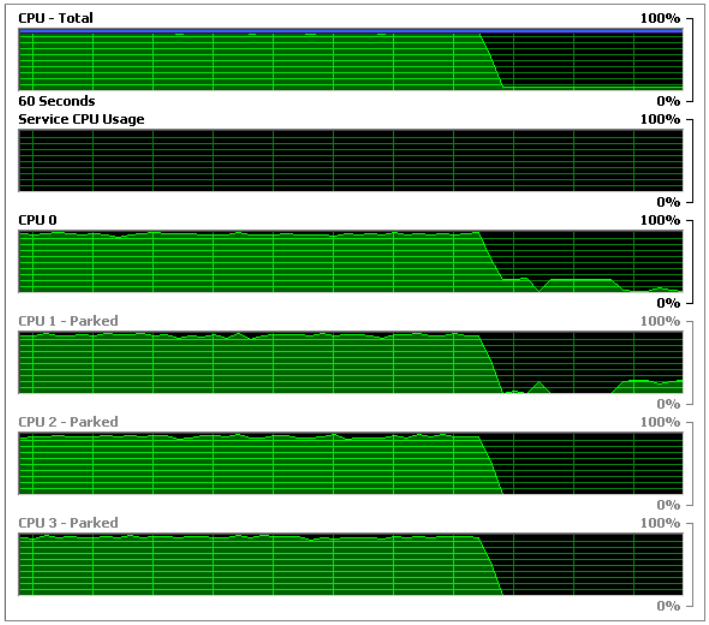
I Figur 3.4 ser vi at sjekken gir en CRITICAL-tilstand på service-objektet, på grunn av et gjennomsnittsbruk av CPU på 96 %.



Figur 3.4: Visning av en CPU-sjekk som gir CRITICAL i Icinga

I Figur 3.5 ser vi at alle de fire CPU-kjernene jobber opp mot maksimalt. Et Batch-script ble kjørt lokalt på serveren som ble overvåket for å generere CPU-bruk.

```
1  ## winloop.bat
2  ##
3  # Runs loop.bat 10 times.
4  #
5  @echo off
6  for /l %%x in (1, 1, 10) do (
7      start loop.bat
8  )
9
10 # loop.bat
11 @echo off
12 :loop
13 GOTO loop
```



Figur 3.5: CPU-bruk for HiG3

På Linux-servere baserer sjekk av CPU-bruk seg på "load-average" [51, 52]. Dette er i hovedsak et gjennomsnitt for hvor mange prosesser som bruker eller venter på CPU-en, men disk- eller nettverks-I/O kan også spille inn. For servere med flere kjerner vil dette fortone seg annerledes da en kan utføre flere prosesser parallelt. Load-tallene må deles på antallet CPU-kjerner for at det skal kunne brukes samme grenseverdier uavhengig av hvor mange kjerner serveren har. Dette gjøres med opsjonen "r".

Tallene som hentes ut er gjennomsnittet for de siste 1, 5 og 15 minuttene. Det er mer interessant hvis load-en er høy over lengre tid, derfor er grenseverdiene lavere for 5 og 15 minutters intervallene. Eksempel på load-tall: **load average**: 0.65 0.42 0.36.

check_load fra nagios-plugins: I service-objektet:

```
check_command  check_nrpe!check_dist_load!0.9,0.7,0.5  1.2,1.0,0.9
```

Kommando brukt i command-objektet check_dist_load:

```
./check_load -r -w $ARG1$ -c $ARG2$
```

3.7.2 Minne

Datamaskiner som bruker opp tilgjengelig minne må skrive til disk for å få plass til mellomlagrede data. Data som må hentes fra disk vil ha en betydelig høyere aksesstid enn når RAM brukes til mellomlagring [53]. Kontinuerlig høyt minneforbruk kan være en indikasjon på flere minnekrevende applikasjoner på samme server, at en applikasjon har minnelekkasje, eller at mengden minne ikke strekker til.

For Windows-servere brukes CheckMem fra NSClient++

```
./check_nrpe -u -H 10.60.0.22 -p 5666 -c CheckMem -a MaxWarnUsed=80%
    MaxCritUsed=90% type=physical
```

Svar:

```
OK memory within bounds.|'physical memory %'=16%;80;90 'physical memory'=1G
    ;4;5;0;6
```

For Linux benyttes plugin-en check_mem [54], da det ikke følger med noen plugin for å sjekke minneforbruk i nagios-plugins.

```
./check_mem.pl -w 80 -c 90
```

3.7.3 Disk

En full harddisk kan skape problemer for applikasjoner som lagrer data og logger fra systemet kan stoppe. Ved høyt minneforbruk og bruk av virtuelt minne i kombinasjon med en full disk, vil ikke harddisken kunne utnyttes som swap, og applikasjoner kan stoppe å fungere.

For Windows-servere benyttes "CheckDisk" fra NSClient++. For å sjekke flere disker brukes opsjonen "CHECKALL", som gir en oversikt over alle diskene på serveren.

```
./check_nrpe -u -H 10.60.0.22 -p 5666 -c CheckDriveSize -a Drive="C"
    MaxWarnUsed=90% MaxCritUsed=95%
./check_nrpe -u -H 10.60.0.22 -p 5666 -c CheckDriveSize -a MaxWarnUsed=94%
    MaxCritUsed=96% CheckAll
```

For Linux-servere benyttes sjekken "check_disk" fra nagios-plugins. Her spesifiseres baner som er montert i filsystemet. check_disk kjøres slik:

```
./check_disk -w 8% -c 4% --mountpoint --all
```

Svar:

```
DISK OK| /=1232MB;15430;17359;0;19288 /lib/init/rw=0MB;402;452;0;503 /dev=0MB
    ;394;443;0;493 /dev/shm=0MB;402;452;0;503
```

3.8 Tjenester

IKT-avdelingen ønsket å overvåke tjenester og prosesser på serverne. Prossesser er instanser av programmer som kjører. Tjenester er prosesser som kjører i bakgrunnen.

Overvåkning av tjenester vil innebære å se på om en eller flere prosesser kjører til en hver tid. NSClient++ har muligheten til å se om en bestemt tjeneste kjører eller har stoppet gjennom "CheckServiceState". Her spesifiserers navnet på tjenesten, og sjekken svarer på om denne tjensten kjører.

```
define service {
    service_description DHCP Service
    hostgroup_name      dhcp_servers
    check_command       check_nrpe!CheckServiceState!DHCPServer=started
        ShowAll
    use                 generic_service
}
```

For Linux brukes plugin-en "check_procs" fra nagios-plugins. For denne plugin-en spesifiseres også hvor mange instanser av prosessen som skal forekomme.

```
define service {
    use                  generic_service
    hostgroup_name       linux_servers
    service_description  NRPE Check my process
    check_command        check_nrpe!check_process!sshd 1:40
}
```

Å se at en tjeneste kjører via Windows kan gi falsk informasjon. Et eksempel på dette er at tjenesten "Microsofts terminal services" står som kjørende i Windows, men brukere får ikke koblet til. Dette kommer av at tjenesten har hengt seg, uten at den står som "stoppet". Dette merkes ikke før brukere ringer inn og beskriver problemet [55]. Som nevnt i 2.2 vil en bedre sjekk være å teste selve tjenesten. Dette gjøres for alle tjenestene beskrevet i de neste delkapitlene.

3.8.1 LDAP, DNS og DHCP

Tjenester som LDAP-autentisering, DNS-oppslag og DHCP-leasing er en viktig del av tjenestene IKT-avdelingen leverer.

LDAP-tjenesten [56] gjør at brukere får logget på trådløse nettverk og autentisert seg for andre tjenester IKT-avdelingen leverer.

DNS-oppslag gjøres hver gang en enhet skal oversette en IP-adresse til et hostname eller omvendt. Uten DNS vil ikke enheten kunne kontakte andre enheter ved å benytte hostname, som brukes i eksempelvis web-adresser [57].

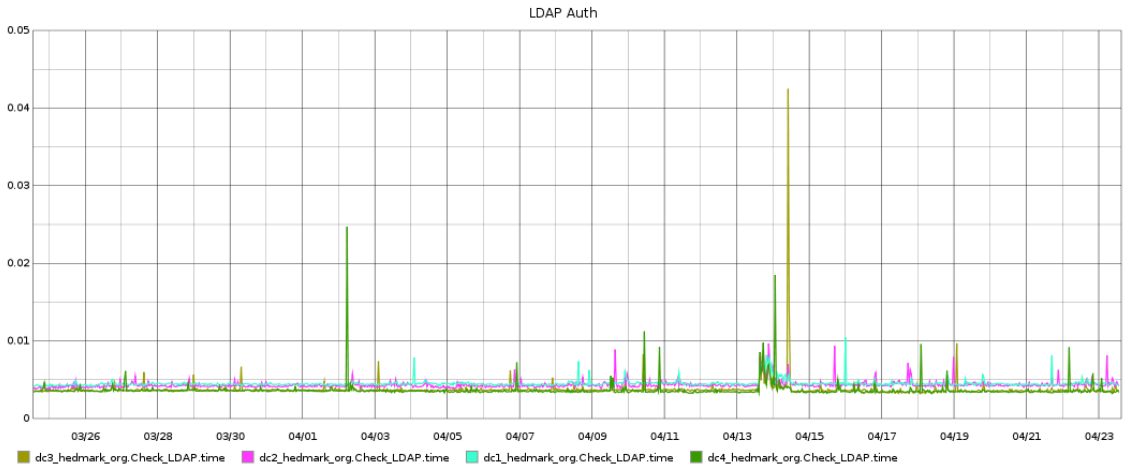
Når en ny enhet kobles til nettverket vil denne få tildelt en IP-adresse av DHCP-serveren. Samtidig får den informasjon om gateway og DNS-servere. Uten dette vil ikke enheten få kommunisert med andre enheter på det lokale nettverket og Internett [58].

Sjekkene for alle de tre tjenestene vil bli gjort direkte fra Icinga-serveren. Denne står i et eget nettverk. Derfor vil det kunne oppstå situasjoner der Icinga rapporterer at tjenestene fungerer, men det fungerer ikke for brukere tilkoblet andre nettverk. En løsning på dette kan være å kjøre sjekkene via en server i hvert nettverk hvor brukere er tilkoblet.

LDAP

For å overvåke LDAP-tjenestene benyttes plugin-en "check_ldap" som følger med i nagios-plugins. Plugin-en kobler til LDAP-tjenesten og prøver å autentisere en spesifisert bruker. Her vil sjekken returnere OK, om den fikk koblet til og autentisert.

Ytelsesdata har blitt samlet inn over 30 dager, for å sette grenseverdier for når Icinga skal gi varsel om treg innlogging. I Figur 3.6 vises det at tjenesten sjekket på fire LDAP-servere, over en måneds periode bruker rundt 0.0044 sekunder på å autentisere. Ut ifra dataen som er samlet inn settes grenseverdien for WARNING til 0.01 sekund, og CRITICAL til 0.02 sekunder.



Figur 3.6: Ytelsesdata i sekunder ved LDAP-autentisering

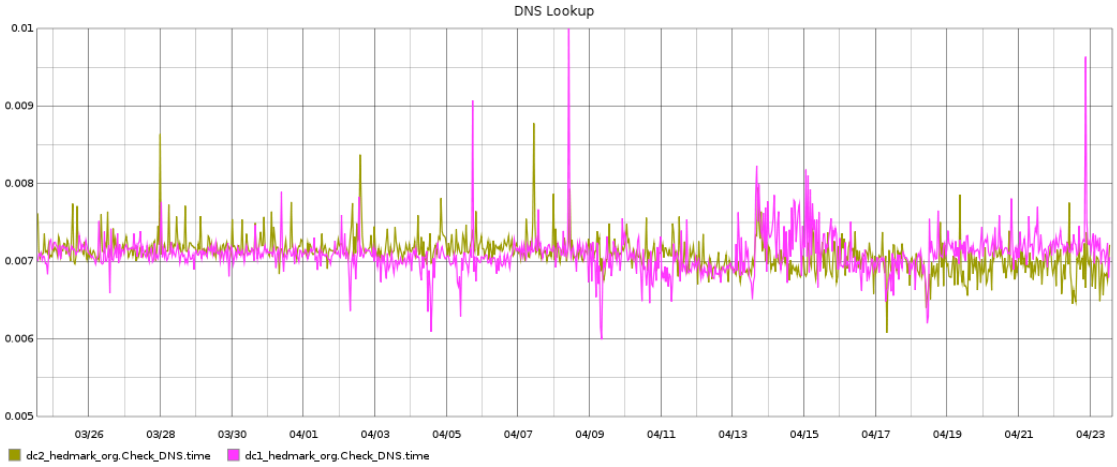
Kommandoen for hvordan sjekken kjøres er vist under:

```
./check_ldap -H 10.60.0.21 -b icingauser -P pass -w 0.01 -c 0.02
```

DNS

DNS overvåkes med plugin-en "check_dns". Denne, på samme måte som check_ldap kobler seg til selve tjenesten. Dette fungerer ved å gjøre et DNS-oppslag på en spesifikk IP-adresse, og verifisere dette mot et satt hostname. Dersom dette stemmer vil plugin-en returnere OK, sammen med ytelses-data på hvor lang tid oppslaget tok.

Figur 3.7 viser data samlet inn fra to DNS servere over 30 dager. Disse dataene viser forventet tid for et oppslag, og ut ifra dette ble grenseverdien for WARNING satt til 0.01 sek og CRITICAL til 0.02 sek.



Figur 3.7: Ytelsesdata i sekunder ved DNS-oppslag

Kommandoen for hvordan sjekken kjøres er vist under:

```
./check_dns -s 10.60.0.21 -H icinga1.monkey.local -w 0.01 -c 0.02
```

DHCP

Overvåkning av en DHCP-tjeneste ble testet i labmiljøet med plugin-en "check_dhcp". Denne sender en DHCPDISCOVER-pakke til DHCP-serveren. Hvis DHCP-tjenesten fungerer får plugin-en en DHCPOFFER-pakke som respons. Dersom denne inneholder en korrekt lease, returnerer plugin-en OK til Icinga sammen med tiden det tok.

Kommandoen for hvordan sjekken kjøres er vist under:

```
./check_dhcp -s 10.60.0.22
```

3.8.2 Webservere

IKT-avdelingen drifter mange webservere som benyttes av, eller gir informasjon til brukere. Plugin-en "check_http" [59] fra nagios-plugins benyttes for å sjekke at tilkoblingen til webserveren svarer på henvendelsen. Deler av forventet innhold sjekkes for å se at websiden ikke har blitt endret av uvedkommende.

Kommandoen for å gjøre dette er for siden http://portal.hedmark.org:

```
./check_http -H portal.hedmark.org -s "Velkommen til Hedmark fylkeskommunes
    portal" -S
```

For websider som skal svare via HTTPS, sjekkes også utløpsdato for SSL-sertifikatet. I eksempelet under vil plugin-en gi en WARNING om sertifikatet utløper innen 50 dager:

```
./check_http -H portal.hedmark.org -C 50
```

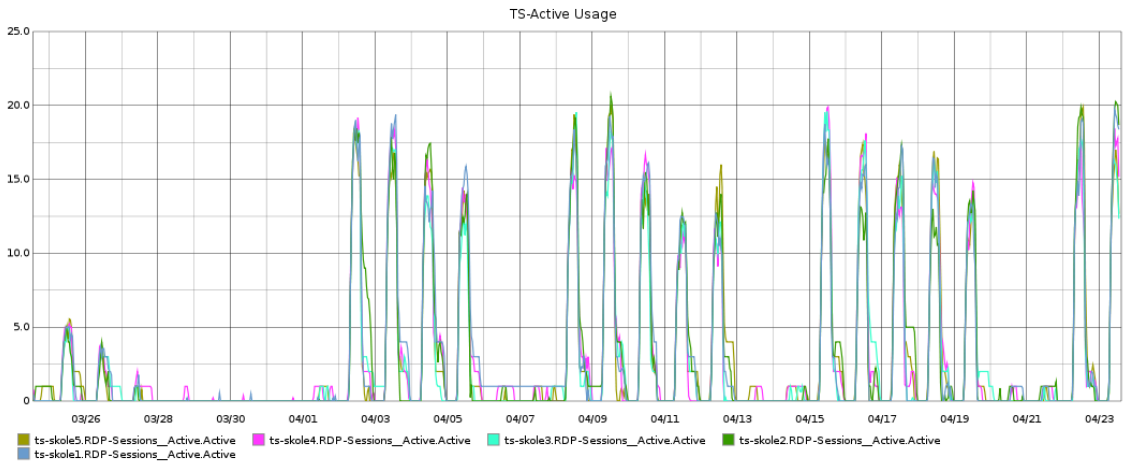
3.8.3 Terminalservere

Mange interne applikasjoner kjøres via terminalservere. Det er viktig at terminalserverne er stabile fordi hver enkelt brukes som en lokal maskin av mange brukere. Ofte kan terminalservere belastes mer i perioder, derfor overvåkes hvor mange aktive brukere som er tilkoblet. Da kan en se hvilke som er mest belastet og når det er behov for å sette inn flere servere.

Antallet aktive og inaktive brukere kan hentes ut fra Windows Performance Counters. Disse overvåkes over NRPE med check_counter i NSClient++. I Figur 3.8 er bruken for en måned uke samlet inn. Ut ifra bruken over en måned er WARNING og CRITICAL satt til 25 og 35 for å varsle unormal bruk.

Kommando som er brukt for å sjekke antallet aktive brukere på en terminalserver er:

```
./command_line check_nrpe -u -H $HOSTADDRESS$ -p 5666 -c CheckCounter -a "
    Counter:Active=\\Terminal Services\\Active Sessions" ShowAll MaxWarn=25
    MaxCrit=35
```



Figur 3.8: Antall aktive brukere på terminalserverne

3.8.4 Microsoft Exchange

Microsoft Exchange benyttes som e-postsystem for fylkeskommunen. Her routes og lagres all e-post som sendes ut og inn av alle brukere. I tillegg benyttes funksjonalitet som kalenderdeling, kontakter, og er integrert med telefonsystemet. Exchange er kritisk for kommunikasjon mellom ulike avdelinger i fylkeskommunen, og utad.

Som for terminalservere, beskrevet i 3.8.3, brukes Microsoft Performance Monitor for å få tilgang til en rekke viktige tall om Exchange-serverenes ytelse. Firmaet SolarWinds, som selger en kommersiell overvåkningsløsning anbefaler annet at følgende parametre overvåkes [60], i tillegg til basisjekkene CPU, disk og minne:

- Antall tilkoblinger. Dersom antallet er høyt kan det føre til at e-postklienter oppfattes som trege av brukerne.
- Gjennomsnittlig responstid på forespørsler. Høye tall vil oppfattes som treghet for brukerne.
- Antall meldinger sendt per sekund. Ved høye tall kan det være mistanke om at mail-serveren blir brukt til spam, eller at klienter på nettverket har blitt kompromittert.
- Antall LDAP-søk som gir timeout. Høye tall kan indikerer mange tilkoblingsfeil mot Active Directory.
- Størrelse på leveringskø. Antall e-poster som ikke har blitt prosessert.

Microsoft har gitt ut egne anbefalinger til grenseverdier som er benyttet som et utgangspunkt [61].

I tillegg til disse sjekkene var det ønskelig å teste hele e-postoppsettet i én sjekk. Til dette benyttes plugin-en "check_email_delivery" [60]. Her sjekkes det at en e-post kan sendes fra SMTP-serveren. E-posten som sendes ut inneholder en unik ID. Videre kobler plugin-en seg til IMAP-tjenesten og sjekker om e-posten med den unike ID-en kom frem. Antall sekunder for hele round-trip-en blir målt. Helst skulle en sendt e-posten fra en SMTP-server som står utenfor nettverket, men det var ikke tilgjengelig under implementasjonensperioden.

Det sjekkes også at websiden for Outlook Web Access er tilgjengelig, på samme måte som andre websider, slik det er beskrevet i 3.8.2.

3.8.5 Redundante oppsett

En ordinær plugin henter status for en service på ett host-objekt. Ved redundante oppsett vil det ikke nødvendigvis være kritisk om en av nodene er nede. For å vurdere statusen til et cluster kan en kjøre en sjekk på hvert enkelt host-objekt, og ta en vurdering basert på resultatene av alle sjekkene samlet.

For å overvåke redundante oppsett, har plugin-ene "check_multi" [62] og "check_cluster" [63] blitt vurdert. check_cluster parser den lokale status.dat-filen og ser hvilken tilstand en service eller en host er ved siste sjekk. Plugin-en check_multi derimot kjører plugin-er mot spesifiserte host-objekter, og en kan definere sammenligningsoperator (<,>,==), som vil bli sjekket mot de returnerte resultatene.

Med check_multi kan en benytte en eller flere egendefinerte kommandoer som parametere. Disse vil parses av check_multi og kan inneholde alt i fra "echo 'Hello, World!', til mer avanserte Perl-script som kjøres ved hjelp av eval. Dette blir brukt for Oracle-databaser som forklart i 3.9.1.

For å evaluere resultatene fra alle sjekkene samlet, kan en definere kriterier som gir et varsel. Da brukes standard Icinga-tilstander, som vist i eksempelet under:

```
command [ HTTP_Node1 ] = check_http -H 192.168.2.10
command [ HTTP_Node2 ] = check_http -H 192.168.2.11
command [ HTTP_Node3 ] = check_http -H 192.168.2.12
command [ HTTP_Node4 ] = check_http -H 192.168.2.13
state [ WARNING ] = COUNT(WARNING) == 2
state [ CRITICAL ] = COUNT(CRITICAL) > 2
```

For check_cluster spesifiseres det om det er et host- eller service-cluster som skal sjekkes. Deretter spesifiseres parametere med navn på host og service, og hvor mange host- eller service-objekter som må ha en gitt tilstand, før det varsles WARNING eller CRITICAL. Fordi check_cluster kjører lokalt på Icinga-serveren, vil den ikke generere noe nettverkstrafikk. Ulempen er at den ikke gir noen informasjon om hvilket host-objekt som er nede eller hvilket host-objekt en service feilet på. Det vil si at den gir bare en overordnet status for det redundante oppsettet.

Under vises et eksempel for kommandoen check_cluster som kjører Check HTTP host-objektene localhost, HiG2, HiG3, og HiG4. Det vil bli gitt en WARNING om en av serverene er nede, og CRITICAL om to er nede.

```
./check_cluster --service -l "Check HTTP" -d $SERVICESTATEID:localhost:Check
    HTTP$, $SERVICESTATEID:HiG2:Check HTTP$ ,$SERVICESTATEID:HiG4:Check HTTP$
     -w @1 -c @2
```

3.9 Databaser

Databaser brukes i mange av applikasjonene IKT-avdelingen drifter for brukerne. På databaseserverne samles dataen fra disse applikasjonene på et sted. Tre forskjellige databasemotorer brukes. Disse er MySQL, MSSQL og Oracle DB. Hver av disse har ulik arkitektur og virkemåte [64], og derfor blir ikke de samme parameterne overvåket på alle. Felles for alle er:

- Connection time. Tiden det tar å koble til SQL serveren.
- Connected users. Antallet aktive sessioner mot SQL serveren.
- Cache hit rate. Antallet spørringer som blir hentet fra cache i et tidsintervall.

Noen av sjekkene er spesifikke for hver databasemotor, og er valgt for å få en oversikt over trege spørringer:

- MySQL: Slow queries. Antall trege spørringer SQL serveren utfører i et gitt tidsintervall.
- MSSQL: Lazy writes. Et høyt tall indikerer at mange minne-page-er må skrives til disk. Dette kan indikere mangel på tilgjengelig minne.
- Oracle: Free table space. Hvor mye ledig plass det er for tabellene.
- Oracle: Log switch interval. Tid mellom hver gang en ny log opprettes for databasen. Et høyt tall kan indikere høy load på serveren.

Connection time

Sjekken av connection time tester om det er mulig å koble til SQL-serveren. Hvis denne sjekken gir en timeout, er det fordi sjekken ikke får kontakt på porten til serveren. Det defineres parametere for hvor lang tid det bør ta å koble til. Hvis tilkoblingstiden blir for lang varsler Icinga om dette. Grenseverdier her er valgt ut fra gjennomsnittlig tilkoblingstid over 30 dager.

Connected users

For connected users sjekkes hvor mange sessioner som er koblet til database tjenesten (per instanse for Oracle DB). Antall aktive sesjoner blir samlet inn om hver enkelt database. Derfor defineres grenseverdier ut fra hvor mange brukere som er koblet til over en periode, og et uvanlig bruksmønster vi gi et varsel.

Cache hit

Cache hit er hvor ofte data hentes ut fra databasemotorens egen cache, slik at dataen ikke leses fra disk. Dette sparer diskene for I/O-operasjoner. Hvis cache hit ligger på et høyt nivå (90 - 100 %), indikerer dette at tabellene det spørres mest mot lagres i cache. Når cache hit ligger under 90 % kan dette være et resultat av at serveren ikke har nok minne til å lagre tabellene i cache. Dette kan indikere et minneproblem.

Oracle mener at cache hit bør ligge på over 90 % [65]. For MSSQL er tilnærmet 100 % cache hit et godt utgangspunkt [66].

MySQL-databasene til IKT-avdelingen lagrer all databasedata i minnet, så her vil det ikke være relevant å sjekke cache hit. Sjekken er satt opp slik at ordinære MySQL-servere vil kunne settes opp med denne sjekken i ettertid.

3.9.1 Forutsetninger for plugin

For at Icinga-serveren skal kunne snakke med de forskjellige databasemotorene trengs en klient for hver av dem.

Oracle

Databaseklienten finnes på Oracle sine nettsider [67]. Den finnes ikke som en deb-pakke, som brukes for Debian. Det finnes derimot en rpm-pakke, som brukes av blant annet Red Hat. Alien ble brukt til å konvertere rpm-pakken over til deb-pakke før installasjon på Icinga-serveren [68].

```
alien oracle-instantclient11.2-basic-11.2.0.3.0-1.x86_64.rpm
dpkg -i oracle-instantclient11.2-basic-11.2.0.3.0-1.x86_64.deb
```

Videre trengs også en databasedriver, som gjør det mulig for Perl å benytte klienten. For Oracle-databaser brukes "libdbd-oracle-perl".

På en Oracle-server vil hver database ha sin egen instans [69]. Derfor vil parametere som overvåkes være forskjellig fra instans til instans. Etter utveksling over e-post med utvikler av plugin-en (se e-post i Vedlegg I), og samtaler med ansvarlig for databaser ved IKT-avdelingen ble det avgjort å overvåke hver enkelt databaseinstanse, med plugin-en check_multi.

Navnene til instansene legges derfor inn som en egendefinert variabel på host-objektet for hver enkelt Oracle-server. Når plugin-en kjøres sjekkes filen tnsnames.ora, som inneholder tilkoblingsinformasjon for hver enkelt instanse, i dette tilfellet må variabelen _HOSTDBINSTANCES inneholde ORA11. Strukturen for en enkelt instanse vises tnsnames.ora vises under:

```
ORA11 =
 (DESCRIPTION =
   (ADDRESS_LIST =
     (ADDRESS = (PROTOCOL = TCP)(HOST = 127.0.0.1)(PORT = 1521))
   )
 (CONNECT_DATA =
   (SERVICE_NAME = ORA11)
 )
)
```

For at Icinga-brukeren skal kunne bruke klienten og finne filen tnsnames.ora kreves det at det brukeren har informasjon om disse i noen miljøvariabler. Dette settes ved å eksportere variablene i "/etc/sysconfig/icinga", som kjøres fra init-scriptet til Icinga.

```
export ORACLE_HOME=/usr/lib/oracle/11.2/client
export TNS_ADMIN=/usr/lib/oracle/11.2/client/network/admin
```

check_multi brukes for å samle en Oracle-servers instanser under samme service-objekt. For eksempel når cache hit-sjekken kjøres, vil check_multi utføre sjekken for alle instansene. I webgrensesnittet vil svarene fra instansene samles under "check cache hit" for hvert enkelt host-objekt. Dette gjør det mer oversiktlig og dynamisk enn en sjekk for hver instans. Under vises konfigurasjonen for dette.

```
define service {
...
    check_command check_multi!check_oracle! -s dbinstances=$_HOSTDBINSTANCES$
        -s host=$HOSTADDRESS$ -s mode=sga-data-buffer-hit-ratio -s warning=93:
         -s critical=90: -s user=$USER5$ -s pass=$USER4$
}

define command {
...
  command_line  check_multi -r 32 -f /etc/icinga/objects/commands/check_multi/
      $ARG1$.cmd $ARG2$
}
```

Da check_multi utfører sin kommando vil "eeval" funksjonen sette sammen en kjede med alle instansene som skal sjekkes, for så å kjøre disse separat. Svaret returnert til Icinga vil inneholde status og ytelsesdata for alle instansene i variabelen "_HOSTDBINSTANCES". Det som blir unikt for denne sjekken er at service-objektet vil inneholde status for alle instansene.

```
eeval [ oracle_health ] =
    my $chain = "";
    foreach my $instance (split(/,/,'$dbinstances$')) {
        $chain .= "-x \"command[ $instance ] = check_oracle_health --connect '
            $user$'\/'$pass$'\@'$instance' --mode '$mode$' --warning $warning$
             --critical $critical$ \" ";
    }
    parse_lines("command [ check_oracle ] = check_multi -r 4 $chain");
```

MySQL

For MySQL ligger de nødvendige pakkene i Debians pakkebrønn og kan installeres med apt. De nødvendige pakkene er "mysql-client", for databasetilkoblingen og "libclass-dbi-mysql-perl", som er en Perl-modul for å kunne koble til en MySQL-server.

MSSQL

For MSSQL var det vanskelig å finne en databaseklient som er fri programvare. Det ble etter noe research besluttet å bruke "FreeTDS" [70], sammen med Perl-modulen "libdbd-sybase-perl".

3.9.2 Plugin

Plugin-ene som blir benyttet for databaser er skrevet av firmaet "Consulting & Solutions" og heter "Check_MySQL_Health", "Check_Oracle_Health" og "Check_MSSQL_Health" [71].

Disse må kompileres fra kildekoden. For å gjøre dette må en først konfigurere de med riktige parametere, før den installeres. Dette er de samme for alle tre plugin-ene.

```
./configure --prefix=/usr/lib/nagios/plugins/ --with-nagios-user=nagios --with
    -nagios-group=nagios --with-perl=/usr/bin/perl --with-statefiles-dir=/tmp
make && make INSTALL
```

For å koble til databaseserverne trengs en servicebruker for hver av de. Denne gis minimale tilganger, slik at denne ikke har tilgang til å endre tabeller og spørre etter info. Brukeren vil kun ha tilgang til å kjøre kommandoer for serveradministrasjon [72].

I MySQL brukes følgende kode for å opprette denne brukeren:

```
GRANT USAGE ON *.* TO 'icinga'@'10.60.0.20' IDENTIFIED BY 'Password';
```

Script for å opprette brukere i Oracle og MSSQL finnes i Vedlegg H.4.

Konfigurasjon for command-objektet for MSSQL og MySQL vises under. $ARG1$ brukes for å sende med hvilken parameter plugin-en sjekker.

```
define command {
...
    command_line   $USER1$/check_<database-motor>_health --hostname=
        $HOSTADDRESS$ --username=$USER5$ --password=$USER4$ --mode $ARG1$ --
        warning $ARG2$ --critical $ARG3$
}
```

MySQL Cluster

MySQL Cluster er et distribuert oppsett for MySQL. Ved IKT-avdelingen benyttes et MySQL Cluster med NDB som lagringsmotor der databasene kjører i minnet. Et MySQL Cluster består av tre forskjellige nodetyper [73]:

- **Management** her konfigureres clusteret og en setter opp hvor mange Data- og SQL-noder som kan kobles til.
- **Data** oppbevarer tabeller og tabelldata i RAM. Disse håndterer lastbalansering, replikering, failover og gjenoppbygging automatisk imellom hverandre.
- **SQL** MySQL-servere som kobler seg til data-nodene for å hente og lagre data.

Den enkleste måten å hente ut statistikk for et MySQL Cluster, er å benytte administrasjonsprogrammet "ndb_adm". Denne kan hentes ut fra installasjonspakken til mysql-cluster [74]. I ndb_adm kan en se hvor mange noder av hver type som er tilkoblet management-noden og minneforbruket til hver av datanodene. De plugin-ene som benyttes baserer seg på output fra ndb_adm.

Antall noder tilkoblet overvåkes med plugin-en check_ndbd [75]. Her ble det gjort en endring i koden slik at serveren som ndb_adm kobler seg til kan spesifiseres som en parameter.

For minnebruk ble det skrevet en egen plugin (Vedlegg H.3), da ingen eksisterende plugin ble funnet som tillater å spesifisere hvilke noder som skal sjekkes. ID-ene til nodene som skal sjekkes ble satt opp som en egendefinert variabel i host-konfigurasjonen. Denne sendes til plugin-en via service- og kommando-objektene. Konfigurasjonen for dette er vist under.

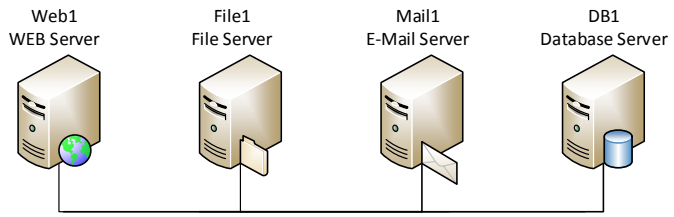
```
define host {
  ...
  _NODEIDS 2,3   ;Data-nodes IDs to check memory usage
}

define command {
  ...
```

```
    command_line    $USER1$/libexec/check_ndb_mem.pl --host $HOSTADDRESS$ --nodes
        $ARG1$ --warning $ARG2$ --critical $ARG3$
}
```

3.10 Applikasjoner

Muligheten for å se om en applikasjon fungerer slik den skal, er en viktig del av overvåkningen. Det er applikasjonene brukerne benytter seg av og vil sende inn feilmeldinger om. En applikasjons tilstand vil bestemmes av flere service-objekters tilstand. I Icinga benyttes et servicegroup-objekt for å gruppere flere service-objekter.

Et praktisk eksempel på dette vises i Figur 3.9. Her vil applikasjonen "Web App for ERP" være avhengig av webserveren for å vise web-grensesnittet til brukerne. En filserver for lesing og lagring av filer, en e-post-server for å sende og motta e-post og en databaseserver som inneholder brukerinfo og andre tabeller.

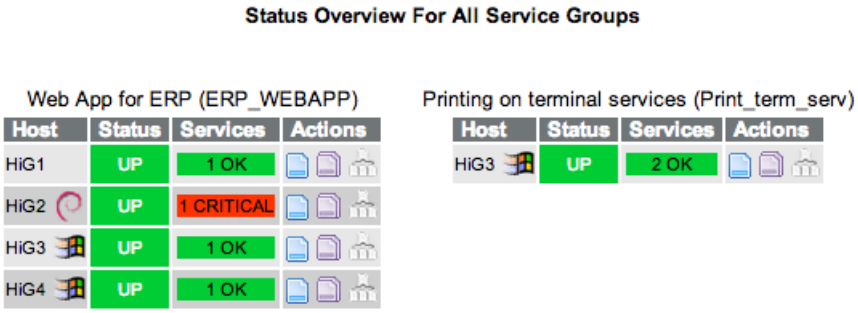


Figur 3.9: Tjenester for servicegroup-objektet ERP_WEBAPP

Konfigurasjonen for dette er vist under. Direktivet "members" setter medlemmene i gruppen der hvert service-objekt refereres til på formen "<host_name>,<service_description>".

```
define servicegroup {
  servicegroup_name ERP_WEBAPP
  alias             Web App for ERP
  members           Web1,Check ERP Web Contents, File1,Check SMB, Mail1,Check
      Exchange, DB1,Check MySQL
}
```

I dette eksempelet kjøres service-sjekker mot alle tjenestene. Service-objektene grupperes i en servicegroup. Et oversiktsbilde over applikasjonen vises i Icingas web-grensesnitt, som i Figur 3.10. Slik blir det enklere for ansatte på servicedesk å kunne gå inn og se hva som er feil med "Web app for ERP" om brukere rapporterer om feil på denne applikasjonen.



Figur 3.10: Oversikt over servicegroup-objekt og status i webgrensesnittet

Det vil det også være naturlig å sette opp servicedependency-er mellom "Check ERP Web Contents" på Web1 og sjekkene for webserveren, filserveren, mailserveren og databaseserveren for å unnga flere varsler ved følgefeil.

3.11 Infrastruktur

Infrastruktur består av de grunnleggende enhetene de andre serverne er avhengig av, og er dermed ekstra viktig å overvåke. I dette prosjektet er det definert til: switcher, routere, brannmurer, UPS, virtualiseringsteknologi og servermiljø.

Det er viktig at design av overvåkningen fører til at man raskt og effektivt skal kunne varsle feil, og hvilke andre enheter dette berører. For å få til dette i Icinga benyttes parent 2.10. De aller fleste enheter i infrastrukturen vil være parent for andre enheter, med unntak av servermiljø. Dette reflekteres også i et statuskart i Icinga, som vist i Figur 2.6.

3.11.1 Switcher

Switcher er nettverksutstyr som jobber på lag 2 i OSI-modellen, datalink-laget. IKT-avdelingen benytter switcher som også har lag 3-funksjonalitet. Det vil si at switchen kan kommunisere over IP. Alle switchene IKT-avdelingen benytter støtter SNMP-protokollen, som brukes til overvåkning.

På switchene overvåkes forskjellige sensorer avhengig av hvilke som finnes. Alle switcher har for eksempel ikke vifter.

Det er laget et generisk oppsett som overvåker temperatur, PSU og viftestatus. Hvilken OID denne informasjonen ligger under varierer fra leverandør til leverandør. For noen produsenter kan denne også være lik.

Dersom en vifte eller PSU rapporteres som defekt vil det rapporteres som en CRITICAL-status i Icinga. For temperatur blir grenseverdiene hentet ut fra SNMP-objektet "ciscoEnvMonTemperatureStatusDescr", der enheten selv rapporterer om temperaturtilstanden er normal, warning eller critical [76].

Switchene IKT-Avdelingen bruker er forskjellige modeller fra leverandørne Cisco, Dell og HP. Disse overvåkes med plugin-ene "check_nwc_health" [77] for Cisco og HP, og check_snmp_powerconnect for Dell [78].

3.11.2 Routere og brannmurer

IKT-avdelingen benytter Cisco ASA- og Cisco PIX-routere for routing av trafikk mellom forskjellige nettverk. I tillegg utfører de oppgaver som pakkefiltrering, NAT og IPsec-tunnelering for VPN.

Ressurser

Plugin-en check_nwc_health brukes for å overvåke sensorer, CPU-bruk og minneforbruk.

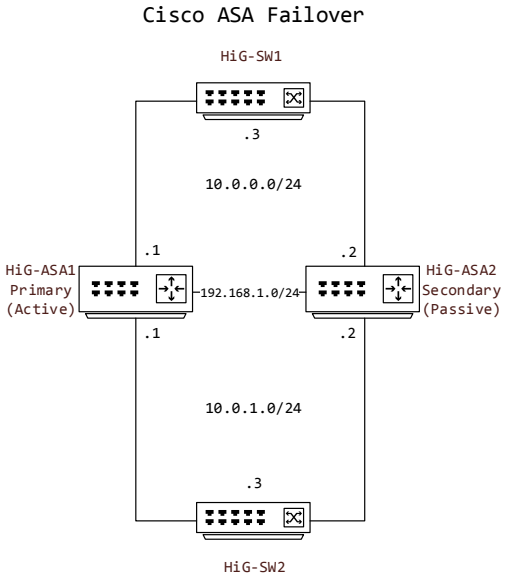
For CPU-bruk sjekkes gjennomsnittet over 5 minutter. Grenseverdier er satt til 80 % på WARNING og 90 % på CRITICAL, i henhold til det Cisco anbefaler [79]. Høy CPU-bruk kan føre til dårligere ytelse, høy rate av buffer-feil og generelle feil med responsivitet [80]

I følge Cisco kan høyt minneforbruk under vanlig drift indikere at brannmuren er under angrep [81]. Dersom en router bruker opp tilgjengelig minne kan det føre til at routeren slutter å svare på kommandoer, telnet-tilkoblinger eller blir mindre responsiv [82]. Cisco anbefaler en grenseverdi på 15 % ledig minne [79]. Grenseverdier for minne er derfor satt til 20 % for WARNING og 15 % for CRITICAL.

Begge brannmurene legges i en hostgroup som er lagt på service-objektet for "cisco_health", slik at lokale ressurser sjekkes på samme måte som switcher. Brannmurene legges også inn i en hostgroup som heter Cisco-failover som benyttes i service-objektet for failover.

Failover

Cisco-routere og brannmurer har en viktig funksjon som kalles failover. Failover fungerer ved at to like enheter settes opp, der en blir satt som "primary" og den andre som "secondary". Secondary kan da ta over for primary dersom det skulle bli nødvendig. Routerne kobles sammen med en seriellkabel. Figur 3.11 viser denne oppkoblingen.



Figur 3.11: Oppsett for Cisco-failover

Dersom secondary mister kontakt med primary, vil secondary ta over nødvendige ruter, brannmurregler og konfigurasjon. Secondary vil dermed overta IP- og MAC-adressen til primary. Primary vil bli satt som passiv (trenger ikke nødvendigvis å ha en IP-adresse på den passive enheten, fordi IP-kommunikasjon kun går gjennom den aktive enheten) helt til en manuelt endrer dette tilbake i konfigurasjonen. [83]

Sjekkene som settes opp for å verifisere at failover-funksjonaliteten fungerer som den skal, og at det ikke har intruffet feil er:

- Hvis primary er satt som aktivt og secondary er passiv returneres OK.
- Om primary er satt som passiv returneres WARNING.
- Om secondary er satt som aktiv returneres WARNING.
- Hvis primary eller secondary får en error returneres CRITICAL.
- Hvis failover ikke er konfigurert returneres UNKNOWN.

Til dette benyttes plugin-en "check_cisco_firewall" [84].

Hvis brannmurene deler management-IP på primary og secondary, er det ingen mulighet for å hente ut informasjon fra den passive brannmuren. Da sjekkes failover-status bare for primary. I et ideelt miljø bør både passiv og aktiv enhet ha hver sin IP-adresse, slik at fysiske feil også kan avdekkes på den passive.

VPN

For brannmurer som har VPN-tjeneste sjekkes antall oppkoblede brukere opp mot det antallet brannmuren er lisensiert for. Det fantes allerede en plugin som sjekker antall tilkoblede brukere [85]. Plugin-en ble endret til i tillegg å hente ut det maksimale antallet, og sammeligne brukt kapasitet i prosent mot grenseverdier, som kommer inn som argumenter. Plugin-en henter ut variablene over SNMP etter OID-er definert i "CISCO-FIREWALL-MIB" [86].

Båndbredde

For å overvåke bruk av båndbredde inn og ut av en port på Cisco ASA og PIX finnes to muligheter: NetFlow eller hente ut verdien fra en teller i et intervall.

Netflow
Funksjonen Netflow [87] fungerer ved at enheten samler inn informasjon og statistikk om alle pakker som går inn og ut av portene. Dette krevet at Netflow konfigureres på alle routerne, og det vil også ta opp ekstra minne og CPU [88]. Det ble derfor avgjort at Netflow ikke skulle benyttes.

Telle pakker
Både Cisco Pix og ASA har tellere for antall byte som går ut og inn på en nettverksport. For å kalkulere bruk av båndbredde over et intervall kan en hente ut disse, vente en gitt periode og hente ut nye verdier. Da vil bruken være ((målepunkt2 - målepunkt1)*8) / antall sekunder.

Mange plugin-er benytter seg av dette ved å lese ut "ifInOctets" og "ifOutOctets" over SNMP. Dette følger Ciscos notat "How to Calculate Bandwidth Using SNMP" [89].

En stor ulempe med å benytte disse er at de er 32 bit, og vil dermed nullstilles kjapt ved høye hastigheter. For 1000 Mbps vil det ta ca. 34 sekunder til telleren har brukt opp tilgjengelig bit og nullstilles. Se Tabell 3.2 for andre hastigheter.

$$\frac{(2^{32}-1)\,B}{\frac{10^9\,b/s}{8\,b/B}} = 34.36\,s$$

Det er derfor anbefalt å bruke 64 bit-variablene ifHCOutOctets og ifHCInOctets i stedet for [90].

Hastighet	**Tid**
10 Mbps	57 minutter og 15.97 sekunder
100 Mbps	5 minutter og 43.60 sekunder
1000 Mbps	34.36 sekunder

Tabell 3.2: Sekunder før en 32 bit teller nullstilles

Plugin-en det er tatt utgangspunkt i for å hente ut båndbreddebruk heter "check_iftraffic64 [91]". Denne er noe omskrevet for å kunne sende inn absolutte verdier som grenseverdier for varsling. I template-en "generic_firewall" er det satt opp to egendefinerte variabler,

"WANWARN" og "WANCRIT", som setter standard grenseverdier for alle brannmurer. For brannmurer der det er normalt med høyere båndbreddebruk, er disse variablene overstyrt i konfigurasjonen for host-objektet.

3.11.3 VMware og Citrix Xen

Ved IKT-avdelingen benyttes virtualserings-teknologier levert av VMware og Citrix. VMware vSphere 5.1 og Citrix XenServer 6.1 benyttes som virtualserings-platformer, og har funksjonalitet som kreves for en virtualisert infrastruktur. På de fysiske serverene kjører hypervisorene VMware ESX 4.1 og Xen 6.1. Hypervisorene (type 1) står for kjøring av virtuelle maskiner og virtualiserer ressurser som CPU, minne, disk, og nettverkskomponenter. De virtuelle maskinene kjører operativsystemene og applikasjoner brukere benytter seg av.

Ressurser som overvåkes for host-ene:

- VMware ESX
  - CPU-bruk.
  - Minne-bruk.
  - I/O.
  - Nettverskort og helse.
- Xen
  - CPU-bruk.
  - Minne-bruk.

For overvåkning av virtuelle maskiner innenfor begge plaformene brukes NRPE. Det overvåkes da som en "egen" server. Ressurser som overvåkes er gjennomgått i 3.7.

VMware

I VMware vSphere brukes begrepene "Datacenter" og "Cluster" for gruppering av host-er og virtuelle maskiner. Datacenter er en gruppering av en eller flere cluster med host-er. Cluster består av flere host-er som igjen kjører virtuelle maskiner. For hver av disse kan det hentes ut informasjon om ressurserbruk.

Plugin

Plugin-en "check_vmware_api.pl" , som er utviklet av fri-programvare firmaet op5 [92], er brukt for å hente ut informasjonen fra VMware VirtualCenter. Denne plugin-en bruker et SDK-bibliotek i Perl [93] for å utføre API-kall til VirtualCenter. Autentisering skjer ved å sende med brukernavn og passord for en brukerkonto opprettet i VirtualCenter-serveren med read-rettigheter. For å utveksle informasjon blir SOAP-protokollen benyttet [94] via HTTPS.

check_vmware_api.pl har mulighet for å hente ut informasjon fra komponentene Datacenter, Cluster, Host, og VM. For hver er informasjon som CPU, minne, I/O, nettverkstrafikk, lagring, og helsetilstand tilgjengelig.

Data blir hentet ut i fra VirtualCenter, der det brukes fire intervaller for lagring av historisk data [95]. Dette er gjennomsnittet for:

1. hvert 5. minutt for en dag.
2. hvert 30. minutt den siste uken.
3. hver 2. time den siste måneden.
4. hver dag det siste året.

Eksempel på bruk av check_vmware_api.pl for å hente ut CPU-bruk for en host:

```
./check_vmware_api.pl -D vc.example.org -u user1 -p password123 -H Example-
    ESX01 -l cpu -s usage -i 300 -T 600
```

Med opsjonen "-i" spesifiseres intervallet en gjennomsnittsverdi skal returneres fra (første intervall i dette tilfellet). Dette gjennomsnittet baserer seg på datapunkter samlet inn for hvert 20. sekund av VMware VirtualCenter. Opsjonen "-T" er "timeshift" i sekunder, som kan brukes for å spesifisere tidsintervallet gjennmomsnittsverdier skal returneres fra.

For de parametere som blir overvåket blir tilleggsopsjonene "-i 300" og "-T 600" sendt med. Etter gjennomgang av output fra plugin-en ble det observert at dette var innstillingene som ga oss gjennomsnittet for det siste 5 minutters-intervallet. Ved bruk av for eksempel "-T 60", som vil si at vi spør 1 minutt tilbake i tid, ble det ikke alltid returnert et resultat siden det ikke har blitt lagret et gjennomsnitt for et 5. minutters intervall i denne perioden.

Andre opsjoner som ble testet før kombinasjonen "-i 300" og "-T 600" var"-i 300", men her kom det 288 resultater tilbake. Dette er antall datapunkt som blir generert i 5 minutters-intervallet for en dag (86400 / 300).

Ressurser

Ressurser som overvåkes nå er CPU-bruk, minne-bruk, I/O responsitivitet, status for nettverkskort, og generell helsestatus som rapporteres fra VirtualCenter.

CPU
Grenseverdier for CPU-bruk er satt til samme nivå som tilsvarende alarm i VirtualCenter, og kan gi en indikasjon på at CPU-kraften ikke strekker til for en eller flere virtuelle maskiner. Dette kan føre til at virtuelle maskiner ikke får kjørt prossesser som applikasjoner benytter. [96]

Teller som blir sjekket er usage.average (%). Dette er aktiv bruk av CPU i prosent iforhold til total tilgjengelig CPU-kraft. Kalkuleringen her er antall CPU-er ganget med klokkefrekvens [97].

Grenseverdier (%): WARNING 75, CRITICAL 90

Minne
Grenseverdier for minne er satt til samme nivå som tilsvarende alarmer i VirtualCenter, og vil gi indikasjon på om virtuelle maskiner krever mer minne enn hosten har tilgjengelig. Når ledig minne går under 6 % (tilgjengelig), kan host-en iverksette

swapping. Når host-en starter å swappe vil den ikke ta hensyn til hva den henter av minne fra de virtuelle maskinene. Dette kan gå utover funksjonalitet og applikasjoner som de virtuelle maskinene kjører.

Counter: usage.average (%). Dette er en prosentandel av konsumert minne iforhold til totalt tilgjengelig fysisk minne [98].

Grenseverdier (%): WARNING 80, CRITICAL 90

I/O
Overvåkning av tid brukt på I/O operasjoner kan avdekke ytelsesproblem mellom host og virtuelle maskiner, og mellom host og lagringsenheter benyttet. Uthenting av informasjon her er foreløpig bare sanntidsdata når sjekken kjører. Counters hentet ut for I/O, aggregeres ikke ved standard innstilling for lagring av data i VirtualCenter. Standard innstilling er "level 1", og det kreves "level 2" for å kunne benytte disse tellerene [99].

Flere anbefalinger eksisterer for overvåkning av I/O tellere [100, 101]. VMware anbefaler at det ikke overstiger 10ms over en periode [102]. Opsjonene "-i 300" og "-T 600" kan benyttes etter aktivering av level 2 for aggregering av data.

Tellere for I/O:

disk.kernelLatency.average - Tiden i millisekunder hver SCSI-kommando bruker fra virtuelle maskiner til den fysiske kontrolleren på host-en. Denne inkluderer også hvor mye tid hver kommando bruker i kø før den blir prossesert, og kan indikere et høyere antall diskforespørsler enn host-en eller den fysiske kontrolleren takler.

disk.deviceLatency.average - Tiden det tar å fullføre en SCSI kommando mellom den fysiske kontrolleren og lagringsenheten benyttet. Dette kan gi indikasjon på ytelsesproblemer for lagringsenheten, eller den fysiske kontrolleren.

Videre overvåkes det totale antallet av kernelLatency og deviceLatency for både skrive- og lese-operasjoner, og disse kan indikerer hvilke diskoperasjoner som skaper ytelsesproblemer. Ved høye tall kan en benytte de to over for å se hvor i systemet ytelsesproblemene befinner seg.

disk.totalReadLatency.average - Total tid samlet i millisekunder for kernelLatency og deviceLatency for skrive-operasjoner.

disk.totalWriteLatency.average - Total tid samlet i millisekunder for kernelLatency og deviceLatency for lese-operasjoner.

I tillegg til de over kjøres det to sjekker som gjenspeiler status fra VirtualCenter. Disse er nettverskort som sjekkes om de er OK, og en generell helsetilstand.

Bruk av NRPE for host-er kan være aktuelt om en vil forsikre seg om at det lagres data om CPU, minne, og diskplass skulle vCenter gå ned. All informasjon om host-er avhenger nå av at VirtualCenter-serveren kjører, som er et "single point of failure".

For kapasitetsmåling og helsestilstanden for et cluster kan det settes sjekker for CPU, minnne og helsetilstanden rapportert fra VirtualCenter. En bruker da opsjonen "-C" med navn på cluster tilsvarende "-H"

Citrix Xen

I motsetning til VMware der all informasjon kan hentes ut fra ett punkt (VirtualCenter), må en for Citrix Xen kontakte en master-host for et gitt cluster. Master-hosten har ansvar for å presentere et administrativt grensesnitt, og videresende kommandoer mot spesifikke host-er i clusteret.

Plugin

For overvåkning av Xen-miljøet ble det kun funnet én plugin som gir muligheten for å hente ut informasjon fra host-er og virtuelle maskiner. Plugin-en "check_xen_api.pl" bruker biblioteket "XenAPI" for å generere URL-er. Disse brukes videre for spørringer mot en RRD-database, hvor "XAPI" lagrer data [103]. Det er ikke like mange tellere som er tilgjengelige som for VMware og I/O-data er foreløpig ikke tilgjengelig for hosts. Ressurser som er satt for overvåkning på host-er er CPU- og minnebruk.

RRD-databasen check_xenapi.pl spør, lagrer data i følgende intervaller:

1. Hvert 5. sekund i en 10 minutters periode.
2. Hvert minutt for de siste 2 timene.
3. Hver time for den siste uken.
4. Hver dag for det siste året.

For hvert 5. sekund blir aktuelle datapunkter lagret. For de andre intervallene blir en minimum-, maksimum-, og gjennomsnittsverdi over gitt tidsperiode lagret ved bruk av en konsolideringsfunksjon. [104]

Et eksempel på bruk av check_xenapi.pl:

```
./check_xenapi.pl -S 10.0.1.50 -u xenuser1 -p password -H XEN_HOST-01 -l cpu -
    s usage -i 300
```

Opsjonen "-H" brukes for å spesifisere hvilken host en vil kjøre sjekken mot. Den må være i samme cluster som oppgitt master-host i opsjonen "-S". Opsjonen "-i" spesifiserer over hvor mange sekunder en skal hente ut data. Resultatet blir da et gjennomsnitt for gitt tidsperiode, og er tilgjengelig for CPU- og minnebruk for host-er.

Som en ser fra eksempelet må det angis addressen til to enheter for å hente ut informasjon, master host-en og host-en en vil ha informasjon om. For å kunne sette en master-IP på ett ojekt, og ikke for hvert host-objekt i et cluster, ble det opprettet et generic-objekt. I generic-objektet settes master IP-en til et gitt cluster via en customvariable. Alle host-objekter som bruker dette generic-objektet arver da denne customvariabelen, og master IP-en kan endres fra et punkt.

Under vises et eksempel hvordan dette kan konfigureres for ett pool med to host-objekter:

```
; generic-objektet
define host {
    use             generic_linux_host
    name            generic_xen_pool_1
    _MASTERIP       10.0.1.50
}

; To host-objekt som arver generic-objektet
define host {
    name            Cluster_1_Xen_Host_1
    address         10.0.1.51
    use             generic_xen_cluster_1
}

define host {
    name            Cluster_1_Xen_Host_2
    address         10.0.1.52
    use             generic_xen_cluster_1
}
```

Opsjenene blir da "-S $HOST_MASTERIP$" og "-H $HOSTADDRESS$".

Det oppstod et problem under implementeringen av check_xenplugin.api, som hadde innvirkning på returnert resultat fra RRD-databasen. Etter å ha lest gjennom koden til selve plugin-en og biblioteket XenAPI som også er skrevet av op5, ble det oppdaget at denne plugin-en bruker siste datapunkt fra intervall nr.1 nevnt over. Resultatet vil da bare være ett datapunkt. Dette er vanskelig å sette varsel på, da en spike kan skje når gitt sjekk kjører. Her ble det avgjort å legge til en ny opsjon for å angi over hvor lang tidsperiode en skal hente ut data. Returnert resultat fra gitt tidsperiode blir gått gjennom, og en gjennomsnittsverdi returneres. Dette er foreløpig bare implementert for CPU- og minnebruk for en host

Ressurser

Ressurser som er valgt å overvåke for Xen hosts:

- CPU-bruk i prosent (totalbruk for alle kjerner ÷ antall kjerner)
  - Grenseverdier (%): WARNING 80, CRITICAL 90
- Minne-bruk (Totalt allokert minne - ledig minne)
  - Grenseverdier (%): WARNING 80, CRITICAL 90

Det må gjøres endringer i check_xenapi.pl for å få korrekt returnerte data, og dette er bare implementer for CPU-bruk og minne for en host slik det står nå.For å minske administrative oppgaver rundt overvåkning av Citrix Xen kan det være aktuelt i ta i bruk check_nrpe for å hente ut informajson om de lokale ressursene. Plugin-en kan derimot brukes for å hente ut informasjon som I/O om virtuelle maskiner, men plugin-en må altså modifiseres til å kalkulere gjennomsnittsbruk.

3.11.4 Trådløse kontrollere

IKT-avdelingen bruker en kontrollerløsning fra Meru Networks på trådløse nettverk. Disse har en viktig rolle i fylkeskommunens nettverk. På en dag har disse rundt 6000 samtidige brukere tilkoblet via aksesspunkter ute på hver lokasjon.

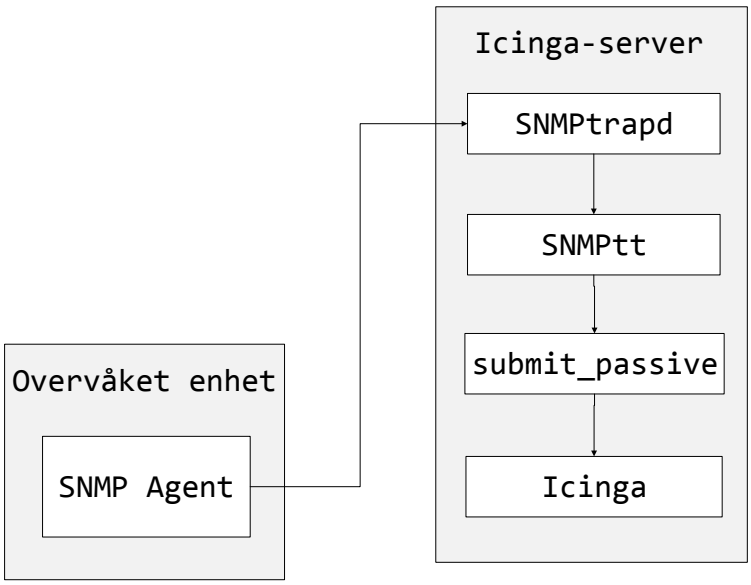
Aksesspunktene inneholder ingen konfigurasjon da de starter opp, denne lastes fra kontrolleren. Alle tilkoblingene går i krypterte sesjoner direkte mot kontrolleren hvor trafikken blir sendt til respektive mottakere. Fordi det går store mengder trafikk over de trådløse nettverkene er det viktig at kontrollerne blir overvåket.

De trådløse kontrollerne mangler støtte for SNMP-get for informasjon som hentes ut fra switcher og brannmurer, som CPU- og minnebruk. Dette vil i følge produsenten komme i en senere firmware-oppdatering. Noe av dette kan løses ved å bruke SNMP-traps i stedet og definere verdier for når disse skal sendes ut direkte på kontrollerne. Her ble relevante trap-meldinger som omhandler kontrollerne satt opp:

- Hardware-feil på kontrolleren.
- Ressursmangel på kontrolleren.
- Rogue AP. Et udefinert aksesspunkt er oppdaget av de andre aksesspunktene.
- Failover til annen kontroller.
- Feil på tilkobling mot Radius.
- Lisens utløpt.

For å få til dette trengs et mellomledd som kan motta SNMP-traps på Icinga-serveren, tolke trap-meldingene og sende informasjonen videre til Icinga.

For å lytte etter SNMP-traps benyttes SNMPtrapd [105]. Her mottas alle trap-meldinger som sendes med riktig community string. Disse sendes så videre til snmptt (SNMP Trap Translator [106]). Her defineres de trap-meldingene en er ute etter, informasjon om OID-ene og kommandoen som skal kjøres når denne mottas. I Figur 3.12 illustreres hvordan en trap blir prosessert.



Figur 3.12: Trap-prosessering

SNMPtt trenger konfigurasjon for alle OID-ene som skal prosesseres. Disse kan opprettes med snmpttconvertmib ut i fra en MIB-fil slik:

```
snmpttconvertmib --in=MERU-WLAN-MIB.my --out=meru.conf --exec='/usr/lib/nagios
    /plugins/libexec/submit_check_result "$aA meru_$N 2 $D"
```

Eksempel på en OID som blir generert i filen meru.conf:

```
EVENT mwlRogueApDetected .1.3.6.1.4.1.15983.3.1.3.13 "Status Events" Normal
FORMAT $*
EXEC /usr/lib/nagios/plugins/libexec/submit_check_result $aA meru_$N 2 "$D"
SDESC
A rogue AP is detected. The AP id, mac address, and other information are
    described in mwlTraContent.
EDESC
```

Her benyttes variabler i snmptt [107]:

- IP-adressen til SNMP-agenten, altså kontrollereren
- Navnet på trap-en
- Beskrivelse på trap fra konfigurasjonfilen.

SNMPtt vil så kjøre kommandoen definert under "EXEC" for OID-en, som i dette tilfellet er å sende informasjonen videre til scriptet "submit_check_result". Dette tar fire argumenter:

- IP-adresse eller hostname
- Service description
- Returkode som setter tilstand på service-objektet (0-3)

- Plugin-output

Scriptet skriver dette sammen med et timestamp til icinga.cmd som Icinga sjekker periodisk etter resultat av passive sjekker.

Service description for alle meru traps som sendes inn vil da være "meru_" + navnet på trap-en. Som plugin-output er det lagt inn beskrivelse av trap-en, da denne som regel er noe mindre kryptisk enn navnet.

Noe å merke seg er at en ikke vil få informasjon når tjenesten er OK igjen, og må sette tjenestene til OK manuelt i Icingas webgrensesnitt.

3.11.5 Servermiljø

For å overvåke temperatur og luftfuktighet på serverrommet ble det kjøpt inn en APC NetBotz 200 med fire eksterne sensorer. Disse ble plassert slik at det er to sensorer på hver side av rack-raden. En vil dermed kunne se temperaturforskjellen mellom forsiden av serverne, ved luftinntak og baksiden, der varmluft går ut.

Hver sensor måler både temperatur og relativ luftfuktighet.

Luftfuktighet

Relativ luftfuktighet er definert som "forholdet mellom partielltrykket til vanndamp, i en gassblanding av luft og vann, og vanndampens metningstrykk ved en viss temperatur". Dette gir en prosentandelen av vann i luften [108].

Mange enheter for overvåkning av servermiljø måler også duggpunkt. Et duggpunkt er temperaturen en viss mengde luft må avkjøles til for at vanndamp skal kondensere. Ved en økning i relativ luftfuktighet vil duggpunktet nærme seg luftemperaturen. Ved 100 % relativ luftfuktighet vil temperaturen og duggpunktet ha samme verdi.

I "Sun Microsystems Data Center Site Planning Guide" anbefales en luftfuktighet på 45 - 50 %. Det meste av datautstyr kan fungere innenfor et bredere intervall enn det, typisk 20 - 80 %. Men de anbefalte verdiene er satt for at det skal være et buffer dersom en har klimaanlegg som kontrollerer luftfuktighet, og det slutter å fungere [109]. Andre, som "American Society of Heating, Refrigerating and Air-Conditioning Engineers" (ASHRAE) anbefaler at den relative luftfuktigheten ikke bør overstige 60 %, mens den nedre grensen er basert på duggpunkt og satt til 5.5° C, der den relative luftfuktigheten vil variere mellom 25 og 45 % [110]. Disse verdiene er også anbefalt av Cisco [111], og omtalt som vidt akseptert.

Høy luftfuktighet kan føre til kondens, som igjen kan føre til korrosjon på komponenter. Ved lavere luftfuktighet øker faren for utladninger av statisk elektrisitet (kritisk ved 30 %), som kan føre til utladninger med ekstremt høye spenningsverdier. Dette kan ødelegge komponenter.

Temperatur

Det er mye uenighet rundt anbefalte verdier for temperatur i serverrom. I et studie utført ved University of Toronto [112], ble det samlet inn data fra tre forskjellige organisasjoners datasentre, deriblant Googles. I studiet ble det konkludert med at faren for hardware-feil ved høyere temperatur øker mindre enn det som har vært vanlig å basere seg på. For DRAM-feil og servere som stopper momentant fant de ingen korrelasjon til temperaturøkning for intervallet i testen (15 - 60 °C). I dette studiet ble det ikke målt eller tatt hensyn til luftfuktighet.

ASHRAE anbefaler en inntakstemperatur 18 - 27 °C, mens Suns anbefaling er 20 - 23 °C. Sun begrunner sin anbefaling med at det er lettere å opprettholde trygge verdier for relativ luftfuktighet med dette temperatur-intervallet. ASHRAE refererer også til studier som viser at det totale strømforbruket kan gå opp ved høyere temperaturer, fordi enheter øker frekvensen på egne vifter [113].

Plugin for overvåkning av temperatur og luftfuktighet i Icinga

Det fantes allerede en plugin for å sjekke sensorverdiene på APC NetBotz [114], men denne sammenlignet verdiene med grenseverdier satt i konfigurasjonen på enheten. Det var ønskelig å samle mest mulig av konfigurasjon på Icinga-serveren, derfor ble plugin-en omskrevet til å kunne ta inn øvre og nedre grenseverdier som argumenter.

3.11.6 UPS

UPS (Uninterruptible Power Supply), ofte kalt avbruddsfri strømforsyning på norsk, blir benyttet til å opprettholde strøm til alle enheter på IKT-avdelingens serverrom dersom strømnettet skulle falle ut. Strømmen filtreres også slik at utstyr ikke vil bli skadet av overspenning. Ved IKT-avdelingen benyttes UPS-er av fabrikatene APC og HP.

For HP og APC overvåkes:

- Gjenstående batterikapasitet.
- Prosentandel belastning.
- Spenning inn og ut.

På APC UPS-er overvåkes også intern temperatur. For både HP og APC overvåkes alle parametere direkte over SNMP med plugin-en "check_snmp" fra nagios-plugins. Her er OID-ene definert direkte i service-objektene.

Under vises et eksempel på hvordan inntaksspenning overvåkes på en APC UPS:

```
./check_snmp -H $HOSTADDRESS$ -C <community string>-o
    .1.3.6.1.4.1.318.1.1.1.3.2.1.0 -l "Voltage In" -w 225:239 -c 220:245 -u "V
    "
```

Objekt	Opsjon	Tilstand	Det sendes varsel om
Host	d	DOWN	Host-objektet er DOWN
	u	UNREACHABLE	En parent er DOWN eller UNREACHABLE
	r	UP	Host-en går til UP
Service	c	CRITICAL	Service-en er CRITICAL
	w	WARNING	Service-en er WARNING
	u	UNKNOWN	Service-en er UNKNOWN
Host og Service	f	FLAPPING	Flapping starter eller slutter
	s	Scheduled downtime	Planlagt nedetid starter eller slutter
	r	UP	Objektet går til UP
	n		Det skal ikke varsles ved noen tilfeller

Tabell 3.3: Oversikt over opsjoner for utsending av varsel

3.12 Varsling

Varsling er en funksjon i overvåkningsløsningen som krever en del finjusteringer og balansering av parametere over tid. Det vil være en balansegang mellom å varsle for mye og for lite. Et tegn på et godt overvåkningssystem er at det er fokusert, og ikke gir overflødig informasjon [115]. Sendes det ofte ut "falske varsel", kan det føre til at alvorlige hendelser blir ignorert, selv om det sendes ut varsler. Når varsel sendes ut bør det kun være til de som trenger å få dem.

3.12.1 Avhengigheter

For å holde antallet varsel som sendes ut lavt benyttes avhengigheter som beskrevet i 2.10 og 2.11. Som tidligere nevnt gjøres dette for å ikke få mer enn en melding ved følgefeil. Parent har blitt benyttet til å reflektere det fysiske nettverksoppsettet. Service dependency-objekter er for eksempel satt på tjenester som blir sjekket via NRPE. Disse vil være avhengig av at NRPE-daemon-en på gitt host kan motta henvendelser.

3.12.2 Konfigurasjon

I Icinga kan en lage kontakter og legge disse i kontaktgrupper. Oppsettet følger samme logikk som at et hostgroup-objekt referer til en eller flere host-objekt i et service-objekt.

Det vil si at contact- og contactgroup-objekter refereres til i et service- eller host-objekt. Disse kontaktene kan bli varslet dersom service-objektet får en annen status enn OK. Dette konfigureres i service-objektet, sammen med hvilke tilstander det skal varsles ved defineres. Dette er vist i Tabell 3.12.2.

Når en feil oppstår vil Icinga vurdere ulike konfigurasjonsopsjoner før det eventuelt sendes ut et varsel. Om konfigurasjonen tilsier at det skal varsles, vil Icinga sende varsler gjennom et command-objekt definert på kontaktene.

3.12.3 Flapping

I Icinga defineres "flapping" som når et host- eller service-objekt skifter tilstand for ofte, noe som resulterer i mange problem- og recoveryvarsler. Dersom detektering av flapping er konfigurert for objektet, vil Icinga stoppe varsler for det dersom prosentandelen flapping overstiger en konfigurert grense. Objektet regnes som å ha startet å flappe når andelen for første gang overstiger en "høy"-grense, og stopper når andelen er under en "lav"-grense igjen. [116]

Denne prosentandelen er kalkulert ved at:

1. resultatet for de siste 21 sjekkene blir lagret.
2. Icinga finner tilstandsendringene fra de lagrede resultatene.
3. Flap-prosent blir kalkulert ut fra andelen tilstandsendringer, der resultatet av de nyeste sjekkene vektes høyere.

Standardgrense er satt til 20 for høy og 5 for lav i icinga.cfg. Grensen kan også konfigureres på hvert host- og service-objekt med direktivene "low_flap_threshold" og "high_flap_threshold", som vist i eksempelet under:

```
define host {
    name                    important_servers    ;template
    use                     generic_host
    flap_detection_enabled  1       ; Flap detection is enabled
    low_flap_threshold      70      ; Stop flapping at 70 %
    high_flap_threshold     80      ; Start flapping at 80 %
    register                0
}
```

3.12.4 Oppsett av kontakter og kontaktgrupper

IKT-avdelingen ønsket å kunne styre kontaktinformasjon og kontaktgrupper fra Active Directory. Det ble bestemt at ansvarsforhold skulle gjenspeiles i kontaktgrupper i møte med IKT-avdelingen(Vedlegg B), for eksempel "ts_ansvarlig" eller "printer_ansvarlig". Icinga støtter ikke integrasjon mot LDAP i konfigurasjonsfilene, men siden disse er rene tekstfiler kan de enkelt opprettes og endres med script. Gruppen var i utgangspunktet skeptiske til å la et script gjøre endringer i konfigurasjonsfiler, da det ble fryktet at dette kunne medføre en konfigurasjon med feil. Konsekvensen av dette ville være at Icinga ikke ville kunne lese ny konfigurasjon.

Icinga har en funksjon som gjør at en kan teste konfigurasjonen etter feil med en opsjon på binærfilen (–verify-config). Icinga vil da ta utgangspunkt i rot-konfigurasjonsfilen (icinga.cfg) for så å sjekke alle konfigurasjonsfilene. Dette gjør at alle konfigurasjonsfilene må sjekkes, og ikke bare de som lages for kontakter. Dermed vil ikke kontakter og kontaktgrupper bli oppdatert dersom det allerede finnes en feil i konfigurasjonen.

Det ble skrevet et Perl-script (Vedlegg H.2) som henter ut kontakter og kontaktgrupper fra en bestemt OU i Active Directory. Perl ble benyttet fordi det ble funnet et godt sett med LDAP-moduler, "perl-ldap" [117] som gjorde tilkobling og søk enkelt. Scriptet kalles fra

Bash-scriptet vist under, som henter resultatet fra synkroniseringen, og sender dette til Icinga. En cron-jobb er satt opp for å kjøre synkroniseringen en gang i timen.

```
#!/usr/bin/env bash
# This script runs a perl script which syncs contacts and contactgroups to Icinga
# it reports the results of the sync as a passive check to Icinga for the service "
    Contact sync"
# MonKey 2013

script="/root/Scripts/ad_sync.pl"
submit="/usr/lib/nagios/plugins/libexec/submit_check_result"
contact_sync=$(perl "$script" --gen-service 2>&1)
exit_code=$?

if [ $exit_code -ne 0 ]; then
        icinga_code=2
        if [ $exit_code -eq 2 ]; then
                icinga_code=1
        fi
else
        icinga_code=0
        contact_sync="Sync OK"
fi

$submit "localhost" "Contact sync" "$icinga_code" "$contact_sync"

exit 0
```

Perl-scriptet som kalles fungerer ved å:

1. hente alle medlemmer av gruppe "icinga_kontakter" og medlemmer av undergrupper rekursivt.
2. det opprettes en konfigurasjonsfil for hver av medlemmene.
3. alle grupper under "Kontaktgrupper" hentes ut.
4. medlemmene av hver gruppe hentes ut og det opprettes en konfigruasjonsfil for hver gruppe.
5. det opprettes en egen service-objekt-template der kontaktgruppen er satt til å få meldinger, slik at tjenester kan arve fra denne.

OU: icinga_kontakter
- Gruppe: alle_kontakter
- OU: Kontaktgrupper
  - Gruppe: ts_ansvarlig
  - Gruppe: exchange_ansvarlig
  - Gruppe: mysql_cluster_ansvarlig

For hver gang en konfigurasjonsfil skrives blir det først tatt en backup dersom en fil med gruppe- eller kontaktnavnet finnes fra før. Etter at den nye filen er skrevet, sjekkes konfigurasjonen og dersom det oppdages feil vil backupen bli kopiert tilbake.

Til slutt sendes resultatet som en passiv sjekk til Icinga. Dersom ingen feil oppstår sendes OK. Hvis ikke vil det gis en CRITICAL der feilmeldingene blir sendt med som resultat av sjekken. Det sjekkes også om e-post-adresse og mobilnummer er definert for alle kontakter. Hvis dette ikke er satt vil det gis en WARNING på service-objektet "Contact sync" i Icinga.

For å sikre at synkroniseringen kjører er service-objektet satt opp med en freshness-sjekk. Dette vil si at det forventes at Icinga mottar et resultat av sjekken hver time (pluss en feilmargin på ett minutt). Dersom det ikke skjer vil det bli satt en feil på "Contact sync". Konfigurasjonen for service-objektet er vist under:

```
define service contact_sync {
    use                     generic_service
    host_name               localhost
    service_description     Contact sync
    active_checks_enabled   0 ; Only use passive checks for this service
    passive_checks_enabled  1
    check_freshness         1
    freshness_threshold     3660 ; 1h + 1 min splay time
    check_command           check_dummy!2 "No contact sync has been run for 24
        hrs"
}
```

Timeperiod

Objektettypen timeperiod er sterkt knyttet til kontakter. Her kan en definere perioder for varsling. En har mulighet til å definere spesifikke dager, datoer eller hver n-te dag.

Det var ønskelig at det ikke skulle sendes ut varsler når IKT-avdelingen har servicevindu, den første onsdagen etter andre tirsdagen i hver måned (dagen etter Patch-Tuesday [118]). For å få til dette ble det først prøvd å definere det slik:

```
define timeperiod {
    tuesday 2 +1        15:00-04:00
}
```

Dette fungerte imidlertid ikke. Løsningen ble å definere perioden for tirsdagen og onsdagen, for så å ekskludere tirsdagen. Slik:

```
define timeperiod {
    timeperiod_name   patch_tuesday
    alias             Patch tuesday
    tuesday 2           00:00-24:00   ;second tuesday of every month
}
```

```
define timeperiod {
    timeperiod_name     service_vindu
    alias               Servicevindu
    tuesday 2 - wednesday 15:00-04:00
    exclude             patch_tuesday
}
```

Eskalering

Når varsel har blitt sendt ut, og tilstanden til et host- eller service-objekt ikke har endret seg over en definert tidsperiode, kan et eskaleringsvarsel sendes. Contact- eller contactgroup-objektene som er referert til i et host- eller serviceescalation-objekt, vil da motta SMS eller e-post om hendelsen.

Et eksempel på når dette kan være nyttig, er at ledere for et driftsteam ønsker å bli varslet dersom temperaturen på serverrommet fortsatt ligger over varslingsgrensen etter at 10 varsler er sendt ut til ansvarlige for kjølingen. Da får lederen mulighet til å ta nødvendige avgjørelser for at problemet skal løses. Konfigurasjonen for dette er vist under:

```
define serviceescalation {
  hostgroup_name          temperature_sensors
  service_description     Check Temperature
  contact_groups          team_leaders
  first_notification      10  ; Ten notifications has to be sent before
      escalation
  last_notification       15  ; After fiftheen notifications, this escalation
      stops. "0" is forever.
  notification_interval   60  ; Escalation notifications are sent every 60
      minutes (default 45)
  escalation_options      c ; Only use escalation when service  is C(ritical)
  escalation_period       op_hours  ; Only escalate during hours of operations
}
```

Det samme kan benyttes på host-objekter:

```
define hostescalation {
  hostgroup_name          domain_controllers
  contact_groups          big_bosses
  first_notification      3
  last_notification       5 ; When does escalation notifications stop
  notification_interval   50
  escalation_options      d ; Only use escalation when host is D(own)
}
```

Varslingsmelding

Selve varslingsmeldingene sendes ut først når host- eller service-objekter er i en hard tilstand (beskrevet i 2.8). Et eksempel er om en brannmurs host-sjekk returnerer DOWN. Da er det ikke ønskelig å varsle over SMS eller e-post, dersom det er snakk om et par ping som mistes. For eksempel kan en sjekk som har returnert DOWN kjøres to ganger til, med et intervall på ett minutt, før en hard tilstand settes, slik:

```
define host {
   use                     generic_host ; Inherit basic config. check_ping
       for host_check
   host_name               hig-fw1
   max_check_attempts      2   ; Number of checks before hardstate
   retry_check_interval    1   ; 60 seconds between each consecutive check
}
```

Dersom objektet oppfyller kriterier for varsling. Det vil si at ingen parent eller avhengigheter er nede vil det sendes ut varslingsmelding.

For et contact-objekt kan det settes opp ett eller flere command-objekter for både host- og service-objekter, som skal brukes for utsending og formatering av varslingsmeldingen.

I eksempelet under brukes command-objektene "host_problem_email" og "host_problem_sms" for varslingsmeldinger:

```
define contact {
    contact_name                    Kari Sysadmin
    email                           kari@example.org
    pager                           480 88 256
    host_notification_commands      host_problem_email, host_problem_sms
    service_notification_commands   service_problem_email, service_problem_sms
}
```

I eksempelet under vises et command-objekt for utsendingen. Her benyttes sendmail [119] til å sende en e-post til en SMS-gateway.

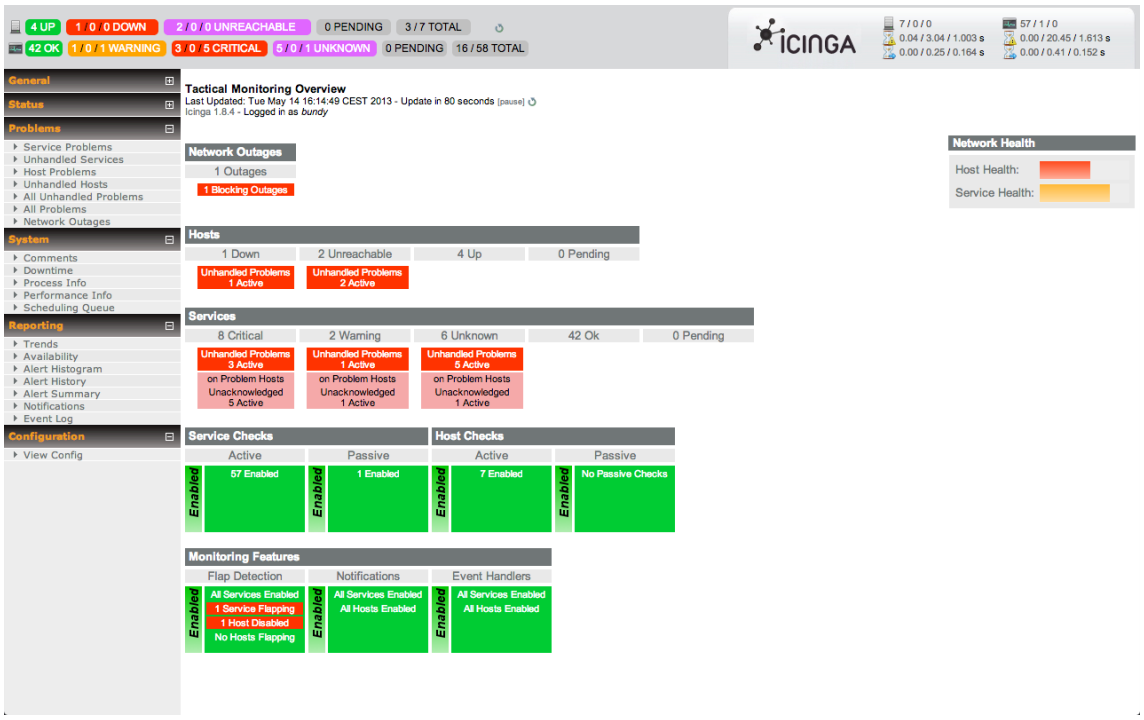
```
define command {
    command_name host_problem_sms
    command_line /usr/bin/printf "%b" "Subject:$CONTACTPAGER$\n***** Icinga
        *****\n\nHost: $HOSTNAME$\nAddress: $HOSTADDRESS$\nState: $HOSTSTATE$\
        nTime: $SHORTDATETIME$\nInfo: $HOSTOUTPUT$" | /usr/sbin/sendmail -f <
        mailfrom@yourdomain> -v $CONTACTPAGER$@<yoursmsgateway>.<tlf>
}
```

Varslingsmeldinger som sendes over SMS vil være korte og lettfattede, mens de som sendes til e-post gjerne er lengre og inkluderer mer utfyllende informasjon. Informasjonen som sendes ut hentes ved hjelp av makroer i Icinga [27].

Alle varslingsmeldinger sendes fra én og samme e-post-server. Dette er ikke helt ideelt, fordi en vil miste både SMS- og e-postvarsling dersom e-post-serveren skulle slutte å fungere. En løsningen på dette kan være å definere en fallback-server som blir brukt hvis Icinga ikke får kontakt med e-post-serveren.

3.12.5 Kritisk varsling

For kritiske tjenester og utstyr er det i konfigurasjonen definert et eget generisk objekt som brukes av de service- og host-objektene som anses som kritiske. Dette gjelder domenekontrollere, nettverksutstyr på hoveddatarommet, temperatursensorer og UPS-er. Disse objektene vil sjekkes oftere enn det som er satt som standard. Varslingsmelding vil også sendes til alle ved IKT-avdelingen om en hendelse skulle inntreffe, også utenom arbeidstid og i servicevinduet.



Figur 3.13: En visning av "tactical overview" fra webgrensesnittet Icinga Classic

3.13 Statusvisning

Som nevnt i 2.5, finnes det et webgrensesnitt for Icinga. Dette er et verktøy for å utføre handlinger på hosts og servicer. Her finnes også et "tactical overview", som vist i Figur 3.13. Det er ikke tilrettelagt for å vise en kort beskrivelse om hva som er galt på de host- og service-objektene med feil, på en stor skjerm. Det er lagt større vekt på administrering av objektene. Det vises et totalt antall objekter med de ulike tilstandene, men det kreves flere handlinger for å få en bedre oversikt. Det ble derfor besluttet å utvikle et webgrensesnitt for statusvisning, som er tilpasset IKT-avdelingens behov.

Ansatte som sitter på servicedesk og besvarer henvendelser vil være hovedbrukerene av statusvisningen. Den skal være et hjelpemiddel for å få et overblikk over tjenester og serveres tilstand.

3.13.1 Design

For at statusvinduet skal imøtekomme IKT-avdelingens ønsker, ble det besluttet å gjøre utviklingen i iterasjoner. Ofte vet ikke brukeren hva de vil ha før de ser et utkast og prosessen har begynt. For utviklingen av statusvinduet har designprosessen beskrevet i ISO 9241-210 blitt fulgt. Der er det beskrevet fire aktiviteter som går i syklus til en får et tilfredsstillende resultat:

- Kartlegge bruksmønstre.
- Spesifisere bruker- og organisasjonskrav.
- Designe løsning.
- Evaluering av designet opp mot kravene.

Utviklingsprosessen startet med et møte med hele IKT-avdelingen for å presentere gruppens forslag til utkast på designet, som vist i Figur 3.14. Her kom det frem forslag om å vise temperaturene fra serverrom-sensorene som en terning der det fysiske oppsettet kom frem.

Eksisterende programvare for denne type fremvisning ble funnet etter research, og det ble besluttet å gjenbruke Nagdash [120]. Løsningen var et godt utgangspunkt for det visuelle, og er skrevet i språket PHP, noe gruppemedlemmene er komfortable med. IKT-avdelingen har også benyttet PHP i tidligere prosjekter.

Nagdash baserer uthenting av informasjon om host- og serviceobjekter på nagios-api [121], som er en tjeneste som må installeres på Icinga-serveren og setter opp et API over HTTP. Icinga har sitt eget API som det var ønskelig å benytte. Uthenting av informasjon måtte derfor endres til å bruke kall til API-et som Icinga Web tilbyr [122].

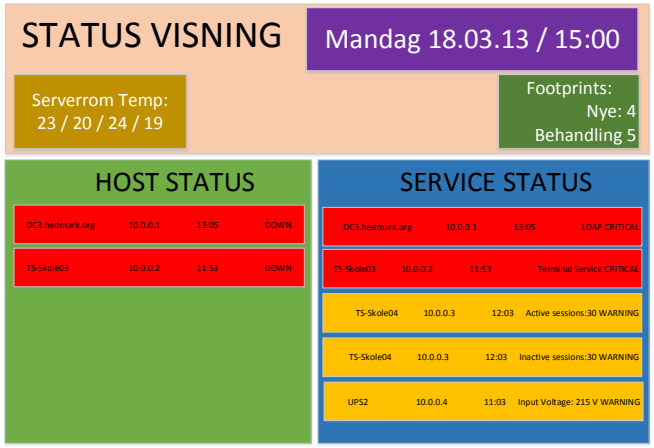
Gjennom to iterasjoner kom det frem flere ønsker som ble lagt inn:

- Uthenting av driftsmeldinger via RSS fra portal.hedmark.org.
- Endre utseendet på “Known Services” og “Known Host problems”.
- Sette kolonnene med hosts og servicer ved siden av hverandre.
- Hente ut og grafe antall åpne, lukkede, og mottatte saker fra sakssystemet Footprints.
- Fjerne irrelevant informasjon, som hvor mange ganger en service-sjekk har fått en soft tilstandsendring.

Det ble også rapportert inn en bug om at datoen på skjermen ikke endret seg før etter en sideoppdatering. Koden som oppdaterer datoen var ikke lagt inn i funksjonen som sørger for automatisk oppdatering. Dette viser viktigheten av å ha brukertesting, da en slik bug kan være vanskelig å oppdage.

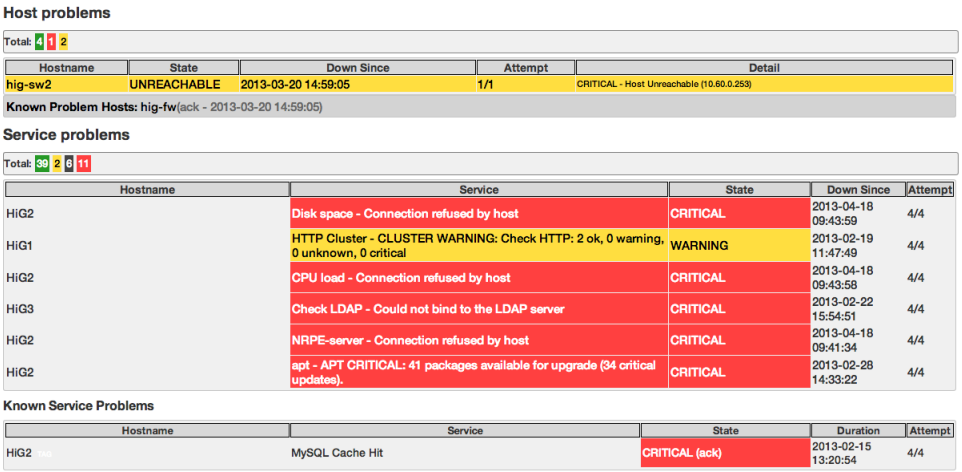
Etter at all funksjonalitet og design var på plass ble all kode gjennomgått av gruppen i fellesskap for å kvalitetssikre løsningen og sørge for at all ikke-triviell kode var kommentert.

I Figur 3.14 vises et utkast til statusvisningen som ble lagt fram for IKT-avdelingen ved første møte.



Figur 3.14: Utkast til statusvindu

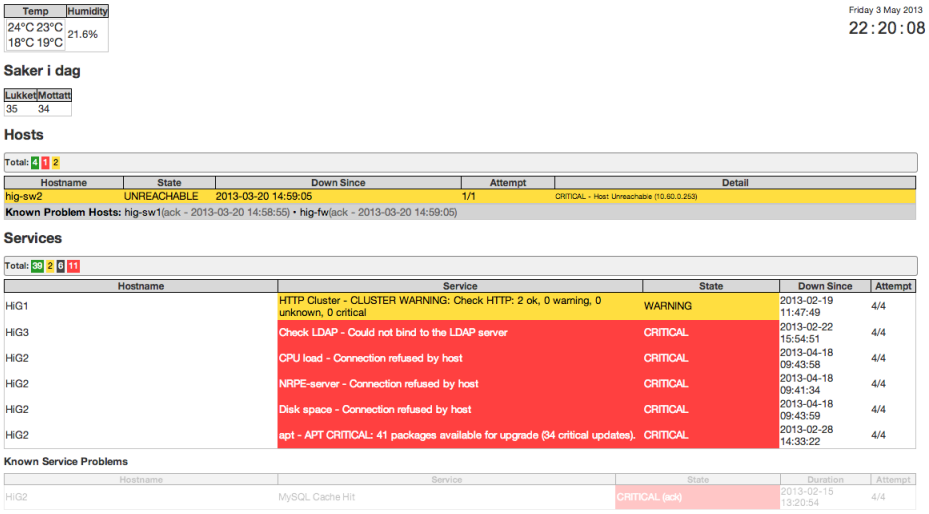
Figur 3.15 viser hvordan Nagdash ser ut med oversatte API-kall:



Figur 3.15: Statusvindu med oversatte API-kall

Figur 3.16 vises statusvisningen etter første iterasjon. Følgende funksjonalitet blitt lagt til:

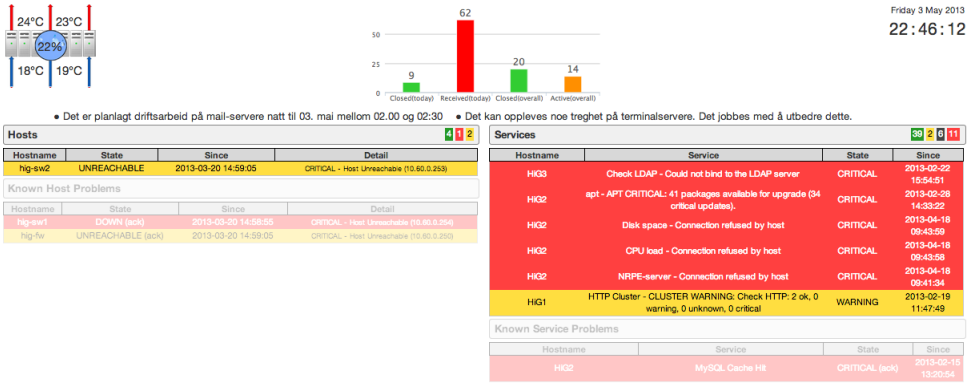
- Temperatur fra serverrommet.
- Antall mottatte, lukkede, og aktive saker fra Footprints.



Figur 3.16: Statusvindu etter første iterasjon

Figur 3.17 viser den ferdige statusvisningen. Følgende funksjonalitet er lagt til fra første iterasjon:

- Visualisering av retning på luftstrøm.
- Temperatur fra de sensorene som er konfigurert.
- Luftfuktighet.
- Kolonnene for host og services er satt ved siden av hverandre.
- Driftsmeldinger fra portal.hedmark.org.
- Grafer over følgende antall saker:
  - Mottatt i dag.
  - Mottatt og lukket i dag.
  - Totalt lukker i dag.
  - Totalt aktive.



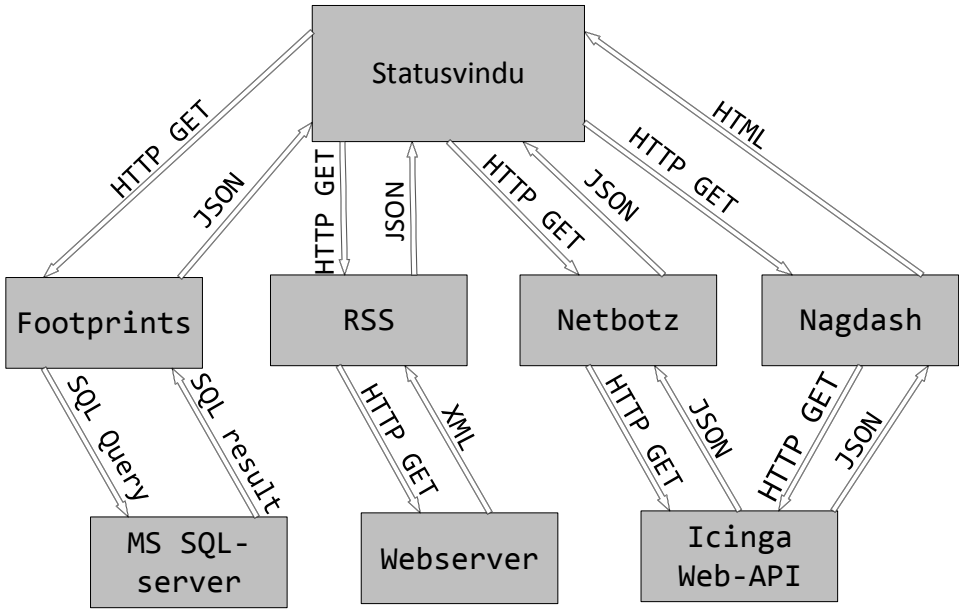
Figur 3.17: Sluttresultatet av statusvinduet

3.13.2 Arkitektur

Statusvindu definerer strukturen for HTML og inkluderer CSS og Javascript.

Hver av modulene "footprints.php" (Vedlegg H.6), "netbotz.php" (Vedlegg H.6) og "rss.php" (Vedlegg H.6) retunerer data ut fra Ajax-kall fra "external.js" (Vedlegg H.6). Dataene bearbeides og settes inn i riktig "div"-element i HTML-koden med Javascript. Nagdash returnerer rå HTML, da denne inneholder tabeller og annen markup i tillegg til dataene.

I filen external.js hentes data fra de ulike modulene og settes inn i statusvinduet. Dette gjøres dynamisk gjennom Ajax, i et bestemt intervall. Hvordan dette henger sammen vises i Figur 3.18.



Figur 3.18: Informasjonsflyt mellom modulene involvert i statusvinduet

3.13.3 Nagdash

For å hente ut data fra Icinga benyttes REST-API-et til Icinga Web. I eksempelet under vises en spørring som henter alle host-objekter som har state-en DOWN (HOST_CURRENT_STATE | = | 1) eller UNREACHABLE (HOST_CURRENT_STATE | = | 2). Viktige parametere:

- <host>: hostnavn eller IP til hosten som kjører Icinga Web.
- <objekt>: objekttypen spørringen skal kjøres mot (for eksempel host eller service).
- filter: Kolonner som skal brukes sammen med de logiske operatorene AND og/eller OR.
- columns: kolonner som skal returneres i resultatet.
- authkey: nøkkelen brukt for å autentisere til API-et.

```
http://<host>/icinga-web/web/api/<objekt>/filter[OR(HOST_CURRENT_STATE|=|1;
    HOST_CURRENT_STATE|=|2)]/columns[HOST_ID|HOST_CURRENT_CHECK_ATTEMPT|...]/
    authkey=<apikey>/json
```

Det returneres JSON-data med kolonnene spesifisert i spørringen.

Det er totalt seks spørringer og de ulike gjør følgende:

1. **hostQuery** henter ut alle hosts som har status DOWN eller UNREACHABLE
2. **serviceQuery** henter ut alle servicer som har en UP host, men en service som er enten WARNING eller CRITICAL
3. **hostTotalQuery** henter ut alle hosts, med en kolonne som forteller hvilken tilstand den er i. Dette blir brukt for å regne ut et antall innenfor hver tilstand.
4. **serviceTotalQuery** samme som 3, men henter alle servicer.
5. **hostPriorityQuery** henter ut en og en host basert på objekt-id. Informasjonen brukes videre for å sjekke om den har en prioritets variabel satt, som vil føre til at den blir prioritert ved fremvisning.
6. **servicePriorityQuery** samme som 5, men henter her ut for en service.

3.13.4 Footprints

For uthenting av antall saker innenfor ulike kategorier i Footprints ble det benyttet SQL-spørringer direkte mot databasen. IKT-avdelingen ønsket å få grafer over følgende antall saker: I footprints.php vil returnert data fra alle de fire spørringene bli slått sammen til et array som videre returneres som JSON-data til external.js via Ajax.

For å generere grafer av returnert resultat i external.js ble jQuery biblioteket Highcarts benyttet. Det ble først testet et annet bibliotek for grafing: jqBarGraph, men dette hadde manglende konfigurasjonsmuligheter og ble valgt bort.

I utdraget nedenfor vises koden for å definere hver søyle i diagrammet (“categories”), og sette de ulike dataverdiene med stats.<variabel>:

```
...
xAxis: {
    categories: ['Closed(today)', 'Received(today)', 'Closed(overall)', 'Active(
        overall)']
}
series: [{
    name: 'Amount',
    data: [
        { y: stats.Closed, color: '#32CD32'},
        { y: stats.Received, color: '#FF0000'},
        { y: stats.ClosedAll, color: '#32CD32'},
        { y: stats.Open, color: '#FF8F00'}
    ]
}]
...
```

Visning av søylene uten et antall for hver kategori, ga bare visualisering av forholdet mellom de ulike kategoriene. Derfor ble det lagt til konfigurasjon for å vise selve antallet over hver søyle, slik:

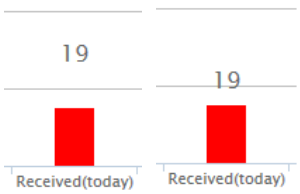
```
...
dataLabels: {
    enabled: true,
    style: {
        fontSize: "16px",
    }
...
}
```

Standard font-størrelse var for liten, så denne ble justert opp. I Internet Explorer 9.0 ble da antallet flyttet for høyt over søylen i forhold til standard høyde. En løsning på dette ble funnet via siden for rapportering av feil til Highcharts [123].

Endring:

```
...
style: {
        lineHeight: "auto"
        fontSize: "16px",
    }
...
```

Figur 3.19 viser høydeforskjellen, der venstre er før endring og høyre etter:



Figur 3.19: Høydeforskjell mellom font over hvert søylediagram i Internet Explorer 9.0

3.13.5 Netbotz

Det er ønskelig at det skal være enkelt å legge til nye sensorer for temperatur og luftfuktighet. Alle sensorene oppdages automatisk av plugin-en som utfører sjekkene for Icinga. Via API-et til Icinga-web hentes siste verdi for denne sjekken ut. Samtidig får en ut hvilken ID hver av sensorene har fått. Dette returneres til external.js som JSON-data. I Javascript settes verdiene inn i statusvinduet og hver sensor kobles til riktig rad.

Utdrag fra external.js

```
...
var front_layout = [2, 1]; // The netbotz sensor IDs for the front row
var back_layout = [5, 4];  // and the back
```

```
4  var columns = Math.max(front_layout.length, back_layout.length); // Largest row is
       used
5  ...
6  var room = new Image(); // Set up background image for the div
7  room.onload = function() {
8      $('#serverroom_climate').css('width', this.width);
9      $('#serverroom_climate').css('height', this.height);
10     $('#serverroom_climate').css('background-image', 'url(' + this.src + ')');
11 };
12 ...
13 // Set the background image to be one with the right number of columns
14 room.src = "image.php?sensor_cols=" + columns;
15 ...
```

Grafikken for servermiljø tilpasses automatisk etter den raden med flest antall sensorer, ved å lime sammen riktig antall bilder dynamisk gjennom GD i PHP (se image.php i vedlegg H.6).

3.13.6 RSS

RSS-feeden hentes inn fra IKT-avdelingens egen drifts-feed, der meldinger til alle brukere legges ut. XML-en fra feeden parses og meldingene sendes til external.js som JSON-data. Dersom det ikke er noen meldinger vil det ikke vises noe. Hvis lengden på feeden overstiger det som får plass på én linje vil den begynne å scrolle bortover. Dette gjøres med jQuery-plugin-en "jQuery Marquee" [124].

3.13.7 Prioritering

Standard visning av et varsel er at det som kom inn sist vises øverst. Ved tilfeller der det er flere varsler, var det ønskelig å kunne gi en prioritering for å få de viktigste varslene til å komme øverst i listen. For å prioritere host-objekter som er DOWN eller UNREACHABLE og service-objekter som har WARNING eller CRITICAL i statusvisningen, ble to løsninger vurdert.

Den ene metoden var å bruke et severity-tall som Icinga kalkulerer basert på hvor mange child-host- og child-service-objekter som hører til et gitt host-objekt som er nede. Dette vises som "network outages" i Icinga Classic der hver host får et "severity"-tall. Det eksisterte ikke noen dokumentasjon på om dette kunne hentes ut via API-et eller hvordan "severity" kalkuleres.

Utrekningen skjer i outages.cgi, som ble funnet etter å lese gjennom HTML-koden og lese kildekoden for outages.c [125]: Child-servicer blir delt på et forhåndsdefinert tall (service_severity_divisor) som brukes for å angi hvor kritisk det er i forhold til child-hosts.

```
temp_hostoutage->severity = (temp_hostoutage->affected_child_hosts + (
    temp_hostoutage->affected_child_services / service_severity_divisor));
```

Det kalkulerte tallet baseres på at hosten som er nede er parent til en eller flere andre hosts. Dette er ikke alltid tilfellet, og metoden vil da ignorere alle host-objekter som ikke

Hosts	**Services**
DOWN (med prioritering)	CRITICAL (med priortering)
DOWN (uten prioritering)	CRITICAL (uten prioritering)
UNREACHABLE (med prioritering)	WARNING (med prioritering)
UNREACHABLE (uten priortering)	WARNING (uten prioritering)

Tabell 3.4: Prioritering av tilstander

er en parent. En annen svakhet er at antall services vil bli kalkulert mot en statisk variabel, som ikke nødvendigvis gjengir viktigheten.

Eksempel:

$$service\_severity\_divisor = 4 \tag{3.1}$$

En host med en child-host som kjører to servicer som er viktige

$$Severity = 1 + \frac{2}{4} = 1.5 \tag{3.2}$$

En host med to child-hosts som kjører to servicer hver, men er ikke like viktige som nr. 1

$$Severity = 1 + \frac{4}{4} = 2 \tag{3.3}$$

Metoden er ikke anvendelig for prioriteringen, da det skal kunne settes på hvilket som helst host- eller service-objekt, og den gjengir ikke viktigheten godt nok.

Den andre metoden gruppen kom frem til var å sette en egendefinert variabel "_PRIORITY" på host- og service-objekter, og å hente ut variabelen via API-et til Icinga Web. Dette ble valgt fordi det kan anvendes på alle service- og host-objekter, og gir et bedre prioriteringsgrunnlag enn severity som vurderer antall child-objekter, noe ikke alle host-objekter nødvendigvis har. En kan heller ikke prioritere en service med metode én.

Implementasjonen av metode to blir gjort ved å spørre API-et for hver host eller service som er hentet ut fra det første kallet i Tabell 3.13.3. Har hosten eller servicen variabelen _PRIORITY definert, vil denne lagres, ellers blir det satt 0 som prioritetstall.

Selve sorteringen av host- og service-objekter blir gjort med PHP-funksjonen usort() med to egenskrevne sammenligningsfunksjoner. [126]. Grunnen til dette er at array-et er flerdimensjonalt, så det må sorteres på verdien til en spesifikk nøkkel. Det skal også sorteres innenfor tilstandene DOWN og UNREACHABLE, og CRITICAL og WARNING. DOWN skal gå foran UNREACHABLE for host-objekter, og CRITICAL foran WARNING for service-objekter, deretter sorteres det etter prioriteringsvariabelen. I Tabell 3.4 vises oppsettet for hvordan dette sorteres.

Kapittel 4

Sikkerhet

Informasjon som brukes av overvåkningsserveren til å avgjøre tilstanden til en enhet eller tjeneste kommer fra eksterne kilder. Plugin-er vil ha eksekveringstillatelse på eksterne enheter, og SNMP trap-meldinger sender informasjon via en port på overvåkningsserveren. Fokus for prosjektet har vært implementasjon, men det er tatt noen sikkerhetsvurderinger underveis. Ulike sikkerhetsrisikoer og hvilke implikasjoner dette har for imlementasjonen vil bli gått gjennom i dette kapittelet.

4.1 Plugin-er

Flere plugin-er har blitt installert i etterkant av Icinga, og dette utgjør en viss sikkerhetsrisiko. Flere plugin-er har et omfang på 1000 - 5000 linjer og det er en tidkrevende prosess å kvalitetsikre alle. De sjekkene som eksekveres på andre enheter, og ikke på selve Icinga-serveren, er plugin-er som kommer med nagios-plugins og NSclient++. Disse blir brukt av et stort antall organisasjoner og andre. Gruppen har også fulgt med på fora og i kommentarfelt for de ulike plugin-ene for å kunne være trygge på de plugin-ene som kjører på hver enkelt host.

De plugin-ene som sjekker en tjeneste eller henter ut informasjon som check_dns, og check_dhcp har i utgangspunktet ingen mulighet for å kjøre direkte på en host. Disse kan utføre ondsinnede handlinger på selve Icinga-serveren. Alle plugin-er som er tatt i bruk kommer enten i fra monitor-exchange.org, op5, opsview, eller nagios-exchange.

4.2 Nettverk

Icinga-serveren står i et adskilt VLAN. Ved hjelp av pakkefiltrering sperres innkommende trafikk fra andre nettverk. Icinga-serveren må kontakte andre servere den skal overvåke og kommunikasjon som er initiert av Icinga tillates i brannmuren.

Utstyr som skal sende SNMP-traps til Icinga må få åpnet tilgang i brannmurene som krysses på ruten. Kommunikasjonen fra Icinga-serveren og ut til andre servere rutes gjennom VLAN, som ikke inneholder kommunikasjon fra periferiutstyr. For at denne kommunikasjonen skal kunne sniffes må disse nettverkene være kompromittert.

SNMP

I implementasjonen er det benyttet SNMP v2c for alt utstyr som støtter det. I noen få tilfeller har bare v1 vært støttet. Både v1 og v2c benytter autentisering basert på en klar tekststring kalt “community string”. Det er heller ingen integritetssjekk som verifiserer at kommandoen som ble sendt er den samme som kom fram. Dette er viktigere dersom en sender “write-kommandoer”, men i dette prosjektet benyttes bare “read” for å hente ut informasjon.

I SNMP v3 er det mange muligheter for økt sikkerhet:

- Konfidensialitet. Kryptering. DES
- Integritet. Hashing med MD5 eller SHA
- Autentisering. Det kan settes rettigheter for spesifikke OID-er for definerte brukere

Overgang til SNMPv3 vil medføre en konfigurasjonsendring på alle enheter som skal overvåkes. En vil også måtte bytte ut eller endre en del plugin-er da disse benytter SNMPv2 eller v1.

NRPE

For kryptering og gjenkjenning av hosts har NRPE implementert OpenSSL med en anonymous Diffie-Hellmann. Plugin-en bruker h-filen dh.h til å generere base-nummer og prim-nummer for diffie-hellmann nøkkelutveksling. Disse blir satt statisk ved kompilering av kildekoden, noe som vil si at det er likt for alle debian-pakkene. Etter autentisering av host og Icinga-serveren brukes AES 256 bit for kryptering av trafikk [127].

NRPE blir som nevnt tidligere i 2.13 brukt for å eksekvere kommandoer på eksterne maskiner, og kan være en angrepsvektor både fra overvåkningsserveren og via andre enheter til de den er installert på. Den eksterne maksinen som NRPE blir installert på vil åpne opp for sikkerhetsrisikoer, men det er fortsatt tilpasninger som kan gjøres for å unngå tilgang for uvedkommende og utførelse av ondsinnede handlinger. I konfigurasjonen må en definere hvilke hosts som skal kunne kommunisere med serveren, en må da vite hvilke IP-er som er definert i denne listen. Her vil overvåkningsserveren i de fleste tilfeller være en definert IP, og da kan det spoofes om denne er kjent for en angriper. En kan også via konfigurasjonen endre porten eksterne enheter skal lytte på. Standardisert port kan finnes i dokumentasjonen så denne bør endres for å skjule tjenestenavn ved portscanning.

Videre utvidelse av implementasjonen av NRPE vil være å ta i bruk forhåndssignerte sertifikater som NSClient++ støtter, det vil da føre til en sikrere nøkkelutveksling enn dagens som bare baseres på et prim- og basenummer som er definert ved kompilering.

I konfigurasjonen for NRPE på Linux-servere brukes direktivet “dont_blame_nrpe=1”. Uten dette vil det ikke tillates å sende argumenter til sjekker som skal kjøres.

NSClient++

For Windows som bruker NSClient++ kan det settes opp passord som i tillegg kreves for kommunikasjon, dette legger enda et lag med sikkerthet. Direktivet “allow_nasty_meta_chars” er satt i konfigurasjonen for å kunne benytte WMI-spørringer som parameter til check_nrpe. Direktivet gjør at spørringene ikke blir regnet som skadelige når NSclient++ mottar de. Dette er fordi en del karakterer vanligvis ville blitt escapet.

I konfigurasjonesfilen for NSClient++ defineres serveradressen slik at kun Icinga-serveren har tilgang til å koble til via NRPE. Dersom denne settes opp til å inkludere et subnett eller inneholder feil adresse vil det være mulig for andre enheter enn Icinga-serveren å koble seg til NRPE-agenten.

4.3 Brukerkontoer

En brukerkonto ble opprettet i Active Directory for Icinga. Denne benyttes ved tilkobling mot domenet på flere områder:

- Sjekk av LDAP.

- Synkronisering av kontakter.
- Tilkobling mot MSSQL database.
- Tilkobling til VMware (Read-only i Virtual Center).

For MSSQL må det også settes tilganger for brukeren. Denne har minimale rettigheter slik at den kun kan lese administrative tabeller.

I MySQL kreves det en bruker i en database på hver server. Disse brukerne får også minimale rettigheter, og har kun tilgang til å lese databasen "INFORMATION_SCHEMA".

For hver Oracle instanse kreves en egen bruker som har tilgang til se logger, statistikk og status for instansenen. Det lages en bruker med tilgang til å gjøre "SELECT"-spørringer på tabellene. Alle brukeropprettingene blitt gjennomgått med databaseansvarlige ved IKT-avdelingen.

Brukeren som brukes for XenServer har rollen Pool Admin, da IKT-avdelingen bruker en versjon som ikke har funksjonen "Role Base Access Control".

4.3.1 LDAP

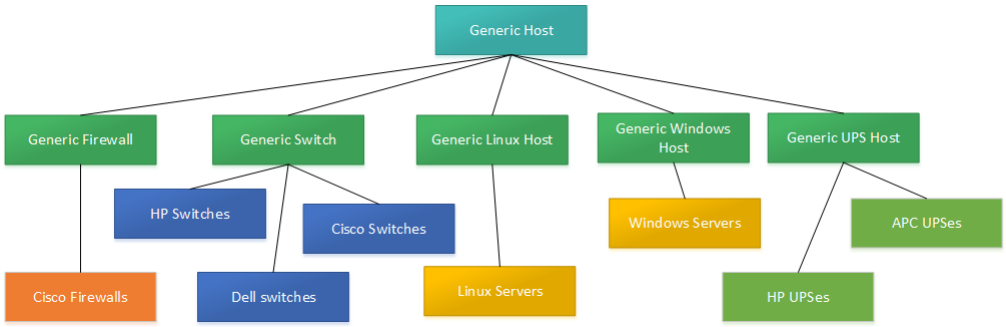
LDAP benyttes ved synkronisering av kontakter og kan benyttes til innlogging i Icinga Web og Icinga Classic, som beskrevet i 5.4. LDAP over SSL kan benyttes for å kryptere denne kommunikasjonen. Da må dette konfigureres og et sertifikat installeres på Icinga-serveren.

Kapittel 5

Dokumentasjon

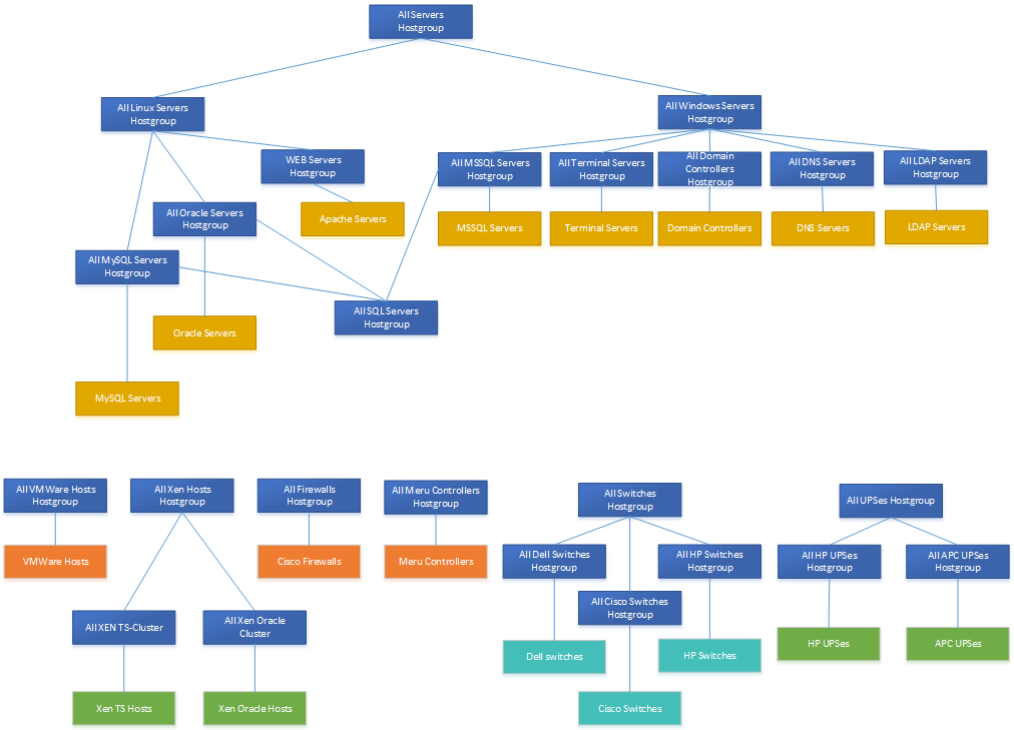
Gjennom prosjektet har det blitt skrevet egen kode, installert ulike pakker, og satt opp konfigurasjon spesifikt for denne overvåkningsløsningen. Dette kapittelet vil gå igjennom ulike komponenter som gruppen anser som nødvendig å forklare grundigere.

I Figur 5.1 vises hvilke generics de forskjellige host-objektene kan ha.



Figur 5.1: Oversikt over host generic plassering

I Figur 5.2 vises hvilke hostgroups de forskjellige host-objektene skal være medlem av.



Figur 5.2: Oversikt over host-objekter og hostgroup-plassering

5.1 Legge til mange host-objekter

I utrullingsfaser vil det være naturlig å legge til mange host-objekter på en gang. Ved å bruke Bash-scriptet "gen.bash", spares tid samtidig som scriptet sørger for at host-objektene er syntaktisk korrekte.

Scriptet genererer host-objekter med atributtene "hostname", "address", "hostgroup" og

"generic". Host-objekter uten hostgroup eller generic supplert, vil få standardverdier definert, mens hostname og address er obligatorisk, og det gis en feilmelding om disse er utelatt.

Eksempelet under viser hvordan scriptet kjøres. Linjenummer skrives ut dersom obligatorisk informasjon mangler:

```
monkey@hig1:~/script$ ./gen.bash servers.csv

    Generating hosts from servers.csv into working directory

    Hostname and Address are mandatory.
    Missing for host on line nr: 4
    Missing for host on line nr: 6
    Missing for host on line nr: 7
    Missing for host on line nr: 8

    Done
    Successfully created 4 hosts
    4 hosts where not created because of errors.
  monkey@hig1:~/script$ ls
  gen.bash  servers.csv  test1.cfg  test2.cfg  test3.cfg  test4.cfg
  monkey@hig1:~/script$
```

Koden under viser hvordan servers.csv er bygget opp. Den inneholder også feil for å vise hva som skjer dersom informasjonen ikke er korrekt:

```
1 10.0.0.1,test1,windows;,generic1
2 10.0.0.1,test2,,jekrjekr
3 10.0.0.1,test3
4 ,,,
5 10.0.0.1,test4
6 1,
7 ,
8 ,
```

Kildekoden til scriptet ligger i Vedlegg H.1.

5.2 Forventet nedtid

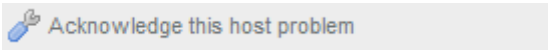
Tilfeller kan forekomme hvor en server må tas ut av drift på grunn av vedlikehold. I Icinga er det ikke nødvendig å fjerne denne fra konfigurasjonen, for å unngå at varslingsmelding blir sendt ut. I både Icinga Web og Icinga Classic har man mulighet til å bruke en funksjon som kalles "Schedule Downtime". Her blir nedetidens start og slutt definert.

I nedetidsperioden vil Icinga nekte varslingsmelding i å bli sendt ut. Når nedetidsperioden er ferdig, eller denne blir utført før avtalt tid, vil kontaktene som skal bli varslet få en melding om at nedetid er over, og Icinga vil ikke lenger holde igjen varsel.

5.3 Stoppe varsling

Når problemet som fører til et varsel for enten et host- eller service-objekt er kjent, kan en legge dette varselet i kategorien "Acknowledged Hosts" eller "Acknowledged Services" ved å benytte en kommando for gitt host- eller service-objekt i webgrensesnittet.

I Figur 5.3 vises knappen for å sette et host- eller service-objekt acknowlegded.

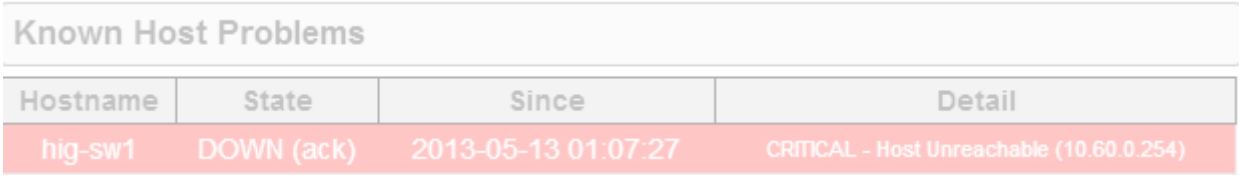


Figur 5.3: Knapp for å settet et varsel som acknowledged

Visning av antallet med enten WARNING eller CRITICAL vil gråtonet om bare et host- eller service-objekt er acknowledged. Tallet vil også flyttes til den midterse kolonnen som vist i Figur 5.4 .

Figur 5.4: Visning av ett acknowledged problem i webgrensesnittet

I statusvinduet vil host- eller service-objektet legges i en egen kategori "Known <objekttype> problems", og ha en gråtonet farge, som vist i Figur 5.5.



Figur 5.5: Separering av acknowledged varsel i statusvinduet

5.4 Autentisering mot Active Directory

LDAP-autentisering brukes for å styre hvem som skal ha tilgang til webgrensesnittene, ved hjelp av grupper og brukere i Active Directory. Dette er støttet direkte i Icinga Web, men må konfigureres på andre måter for Icinga Classic. Det kan opprettes en sikkerhetsgruppe i AD som inkluderer medlemmene som skal få tilgang. Alle som er med i denne gruppen vil få tilgang til Icinga Web, der det også kan defineres hvilken informasjon hver bruker skal se.

Selve LDAP-binden gjøres via PHP-modulen "php-ldap". Når en bruker logger inn sjekkes først AD, dersom brukeren ikke finnes der, vil Icinga-web prøve sin egen database med brukere. Hvis brukeren finnes her og passordet er riktig vil brukeren bli logget inn. Når brukeren har logget inn for første gang via AD, vil brukeren bli lagret i Icinga Web-databasen. Om brukeren endrer passord i Icinga Web vil uansett AD først bli spurt før Icinga Web spør om passordet er riktig i sin egen database

Icinga Classic autentiserer mot AD gjennom modulene ldap og authnz_ldap til Apache2.

Konfigurasjon for Icinga Classic, i /etc/apache2/conf.d/:

```
    AuthName "Authentication"
    AuthType Basic
    AuthBasicProvider ldap
    AuthLDAPURL"ldap://10.60.0.22:3268/dc=monkey,dc=local?samAccountName?sub?(
        objectClass=person)"
    AuthLDAPBindDN "<user>"
    AuthLDAPBindPassword "<password>"
    require ldap-group CN=icinga-login, OU=icinga, DC=monkey, DC=local
```

Konfigurasjon for Icinga Web, i /usr/share/icinga-web/app/modules/AppKit/config/auth.xml:

```
1   <ae:parameter name="ldap_allow_anonymous">false</ae:parameter>
2       <ae:parameter name="ldap_dsn">ldap://10.60.0.22</ae:parameter>
3       <ae:parameter name="ldap_start_tls">false</ae:parameter>
4       <ae:parameter name="ldap_basedn">DC=monkey,DC=local</ae:parameter>
5       <ae:parameter name="ldap_binddn">icingawebauth@monkey.local</ae:parameter>
6       <ae:parameter name="ldap_bindpw"><![CDATA[Password]]></ae:parameter>
7       <ae:parameter name="ldap_userattr">sAMAccountName</ae:parameter>
8       <ae:parameter name="ldap_filter_user"><![CDATA[(&(sAMAccountName=
            __USERNAME__)(memberOf=CN=icinga-login,OU=icinga,DC=monkey,DC=local))
            ]]></ae:parameter>
9   </ae:parameter>
```

5.5 Kontakter og kontaktgrupper

For å opprette en ny kontakt i Icinga meldes brukeren inn i gruppen icinga_kontakter (hvordan dette fungerer er forklart i 3.12.4). Den vil trenge å ha e-postadresse og telefonnummer satt.

Kontaktgrupper hentes fra OU-en icinga_grupper der alle grupper hentes ut og opprettes i Icinga. For at en gruppe skal varsles må dette legges til på et service-objekt. For å slippe å sette opp kontakter for hver service kan en sende med argumentet "gen_service", for å generere en template som kan benyttes på tjenester som sorterer under hver av kontaktgruppene som blir opprettet.

Eksempel på generisk service som blir opprettet:

```
define service {
    name            network_services
    register        0
    use             generic-service
    notification_interval   30
    notification_period     24x7
    notification_options    w,c,r
    contact_groups          network_contact_group
}
```

Denne brukes på en Cisco firewall:

```
define service {
    service_description Cisco Firewall CPU Load
    use             network_services
    name            cisco-firewall-cpu-load
    hostgroup_name      all_firewalls
    check_command       check_network_component!cpu-load
}
```

5.6 Backup-rutiner

I Tabell 5.6, vises viktige filer og mapper for overvåkningsløsningen. Disse bør integreres i en plan for backup av Icinga-serveren.

Filbane	**Inneholder**
/etc/icinga/	Alle konfigurasjonsfiler for Icinga
/usr/lib/nagios/plugins/	Alle plugin-er
/etc/nagios-plugins/config	Konfigurasjon for default plugins
/root/Scripts	Lokale script.
/etc/snmp	Konfigurasjon for snmptt
/etc/default/snmp	Konfigurasjon for at snmpd starter snmptrapd
/etc/sysconfig/icinga	Konfigurasjon for environment variabler
/usr/lib/oracle/11.2/client64/network/admin/tnsnames.ora	Konfigurasjon for oracle-instanser
/etc/init.d/carbon-cache	Init script for grafprosessering gjennom graphite
/etc/init.d/metricinga	Init script for prosessering av perfdata fra Icinga
/opt/graphite/	Konfigurasjon og data for genering av grafer

I tillegg bruker Icinga Classic og Icinga Web hver sine databaser, kalt “icinga” og ”icinga_web”. Graphite bruker også en database, som heter “graphite”. Alle disse kjøres av MySQL lokalt på Icinga-serveren, og bør også inkluderes i en backup.

Kapittel 6

Avslutning

6.1 Evaluering

6.1.1 Gjennomføring av prosjektet

Da prosjektarbeidet ble satt i gang ble det tidlig registrert at oppgavebeskrivelsen var noe løs. Det var derfor usikkerhet rundt hvor omfattende overvåkningsløsning oppdragsgiver ønsket seg. Dersom kommunikasjonen mellom gruppen og oppdragsgiver hadde vært grundigere fra starten av, og kravspesifikasjonen hadde vært mer konkret, ville dette ført til mindre tid på oppklaring ved senere tidspunkt. Det ville også gitt et større overblikk over hva som skulle gjøres. På grunn av dette ble overvåkningsløsningen mer omfattende og tidkrevende enn først forutsett av gruppens medlemmer.

Fordi vi ønsket å dekke flest mulig av kategoriene ulikt utstyr og tjenester IKT-avdelingen drifter i produksjon, valgte vi å sitte på fylkeshuset to dager i uken. Det ble derfor brukt mye tid på reising til og fra fylkeshuset.

I forhold til det Gantt skjemaet som ble satt opp i forprosjeketet har vi fulgt de iterasjonene som ble satt opp, med noen unntak. Servermiljø ble flyttet til etter varsling, da enheten for å overvåke servermiljøet kom senere enn forventet. Iterasjonene Servermiljø og Statusvindu ble utført parallelt, grunnet mindre arbeidsmengde enn forventet i Servermiljø. Det ble også satt opp sjekker for UPS-er i infrastrukturiterasjonen, som i utgangspunktet var planlagt for Servermiljø. Noen tjenester har tatt lenger tid å implementere enn planlagt og disse har blitt jobbet med samtidig som andre iterasjoner har pågått.

Innenfor hver iterasjon var det satt opp faser med planlegging, implementering og en fase med testing og evaluering. Utover i prosjeket ble ikke dette fulgt. Planlegging og evaluering har skjedd mer individuelt for hver enhet og tjeneste som har blitt lagt til i overvåkningen. For alle iterasjonene har det vært møter med statusoppdatering, som planlagt.

Det ble også mye nødvendig dobbeltarbeid med å implementere sjekker i labmiljøet først og så i produksjon. For noen tjenester var det ikke mulig å teste i labmiljøet først.

Valget av Icinga har ikke hindret oss i å få til det som var ønsket. Arkitekturen Icinga baserer seg på er en enkel kjerne hvor tilleggsfunkjsonlitet må legges til. Dette gir stor frihet, og fleksibiltet fordi en kan utnytte et stort antall tredjeparts plugin-er som eksisterer, eller utvikle plugin-er etter behov.

En overvåkningsløsning basert fri programvare kan ha visse ulemper. Ofte kommer slike løsninger uten support-avtaler, noe en som oftest finner i en proprietær løsning. Properitære løsninger markedsføres, og selges, ofte som en komplett pakke der en får produktet sammen med support-avtaler.

Tredjeparts kode kan variere i standard og kvalitet. Ofte har dokumentasjonen vært mangelfull eller ikke eksisterende. Det har ikke alltid vært mulig å finne en plugin som fyller alle behov og da har modifisering av eksisterende løsninger vært nødvendig. Dette krever både tid og noe programmeringskunnskaper.

6.1.2 Organisering av arbeid

Innad i gruppen har det blitt avholdt daglige møter for å oppdatere resten av gruppen om utført arbeid, og om eventuelle utfordringer som har oppstått.

Rollen som gruppeleder har i liten gra vært benyttet. Ved uenigheter og avgjørelser har gruppen alltid kommen til enighet i felleskap. At det ble satt en egen kommunikasjonsansvarlig har etter gruppens mening vært positivt. Det har fungert bra å ha et kontaktpunkt for kommunikasjon utad.

Kontakten med veileder har fungert godt med ukentlige møter og hjelpsomme forslag relatert til arbeidet.

Avtalte møter med oppdragsgiver ble til tider ikke avholdt, grunnet at oppdragsgiver ikke alltid var til stede da gruppen var på Fylkeshuset. Fordi oppsatte møtetidspunkter ikke alltid passet, ble gruppen og oppdragsgiver/teknisk kontakt enig om muntlige oppdateringer utenom møtene. Det var dessuten noe vanskelig å få avtalt overføring av funksjoner fra lab til produksjon med de som hadde ansvar for systemet, da disse ikke alltid var tilstede da gruppen var på fylkeshuset.

Utgangspunktet for utførelsen av prosjektet var at det meste av vårt arbeid skulle skje i et labmiljø. Vi skulle deretter legge fram et forslag til et overvåkningssystem for implementering ved IKT-avdelingen.

Det positive med å ta en større del av implementeringen av overvåkningssystemet er at det er givende å levere en større løsning til oppdragsgiver, og at systemet som er konstruert kan iverksettes slik det nå fremligger.

Ved utvikling av statusvindu endte vi opp med å skrive om nesten all koden i Nagdash. Her gikk det med mye tid, og strukturen på koden ble heller ikke slik gruppen hadde gjort det om det ble skrevet fra bunnen.

Gjennomgang av ulike plugin-er viste seg å ta mer tid enn forventet. Årsaken var for det meste dårlig dokumentasjon og varierende kodekvalitet. Et tilfelle var check_xenapi.pl som ble gjennomgått for å sjekke om det var noe gjennomsnittskalkulering på returnert resultat, og om det var mulig å legge dette til. Plugin-en bruker 12 ekstra Perl-filer for å

sette opp kall til RRD-databasen, og gjennomgang av disse tok for mye av tiden i forhold til gevinsten med å få den implementert. Det var den eneste plugin-en som hadde mer funksjonalitet enn å sjekke statusen for en Xen-host, så det ble priortert å gå gjennom koden for denne. Det har i ettertid lært oss at det ikke alltid vil være besparende å gjenbruke kode. Noen ganger kan det faktisk lønne seg å finne opp hjulet på nytt.

Arbeidsfordeling

Arbeidet har blitt fordelt slik at den av gruppens medlemmer med størst interesse og/eller kunnskap om et tema har satt seg godt inn i dette. Dette har ført til besparing av tid, da det for det meste har blitt utført tre oppgaver parallelt. Dette var nødvendig for å få tid til å gjennomføre målsetningen som ble satt. Utfordringen med denne fordelingen har vært å oppnå god oversikt over det totale arbeidet for hver av oss. Det tok tid å sette seg inn i problemstillinger når andre medlemmer støtte på vanskelige valg eller problemer, og det var behov for å ta felles avgjørelse.

Jevnlig og god kommunikasjon innad i gruppen har resultert i et godt samarbeid med få problemer.

Prosjekt som arbeidsform

Å utføre et prosjekt for en ekstern organisasjon har vært givende og lærerikt. Det å levere et etterspurt produkt til arbeidsgiver gir et godt innblikk i hvordan arbeidslivet fungerer.

Mer tid og ressurser ble brukt på utføringen av bacheloroppgaven enn det som ville blitt med gjenomsnittlige emner med tilsvarende studiepoeng. Dette gjorde at det var utfordrene å følge opp andre fag i tillegg til bacheloroppgaven.

Subjektiv oppfatning av bacheloroppaven

Utføringen av bachelorprosjektet har gitt oss mye lærdom om omfattende prosjekter. Dette er relevant når vi nå skal ut i arbeidslivet. Den type erfaring hadde vi ikke tilegnet oss dersom tiden ble brukt i en forelesningsal, og dermed var dette bachelorprosjektet en godt egnet avslutning på utdanningen.

6.2 Videre arbeid

I denne prosjektoppgaven har hovedfokus vært å bygge opp en overvåkningsløsning som tilfredsstiller kravene i oppgavebeskrivelsen. Det er tatt høyde for tilleggsfunksjonaliteten fra oppgavebeskrivelsen. Hovedsaklig går dette på historisk overvåkning. Det er satt opp grafing av ytelsesdata fra Icinga, men her er det mange muligheter for forbedring og videre arbeid. Dataene kan for eksempel analyseres for å sende varsler til Icinga om trender som indikerer problemer i nær fremtid, for eksempel om bruk av lagringsplass på en filserver.

Parallelt med prosjektarbeidet har IKT-avdelingen gjennomført et prosjekt der en CMDB har blitt implementert. Integrasjon mot denne med konfigurasjon fra objektene i overvåkningen og etterhvert mulighet for å definere avhengigheter som gjenspeiler seg i overvåkningen, kan være aktuelt.

Icinga 2 [128] er under utvikling og er en parallell utviklingsprossess med Icinga 1.x. Planlagt stable release er i slutten av 2013. Icinga 2 er en ny kjerne som skal skrives fra bunnen av og fjerner det som Icinga 1.x har arvet i fra Nagios. Den nye kjernen vil fungere sammen med Icinga 1.x gjennom et kompabilitetslag [129] som oversetter kall. Her kan enten migrering til Icinga 2, oppsett av Icinga 2 sammen med eksisterende Icinga 1.x, eller å evaluere Icinga 2 som et nytt verktøy være aktuelt videre arbeid.

Varsling er implementert slik at statusvisning blir oppdatert og det sendes ut varslingsmeldinger til en eller flere personer over e-post og SMS. Det må altså manuelt opprettes en sak i sakssystemet Fooprints. Her kan det være aktuelt å implementere en automatikk i at ønskede varsler registreres i sakssystemet, med ulike parametre som tjeneste eller host, prioritering, og ansvarlig person.

Asterisk og Trio er to telefonsystemer IKT-avdelingen drifter. Her vil det være relevant å overvåke køer, og annen informasjon. Å integrere dette med Icinga kan være en spennende utfordring å basere en oppgave på.

IKT-avdelingen ved Hedmark fylkeskommune har flere lokale IKT-avdelinger, som i hovedsak løser lokale problemer ved skoler, men som er avhengig av sentrale tjenester som trådløse nettverk og Internett-tilgang. Icinga Web muligheter for tilgangstyring på brukernivå for visning av informasjon. Et mulig prosjekt kan være å gjøre de tilpasningene som må til for å integrere de lokale avdelingene i overvåkningsløsningen.

6.3 Bidrag til fri programvare

Både scriptet for å sjekke hvor mye minne en MySQL Cluster node har brukt og scriptet for å synknronisere kontakt-objekter fra Active Directory ble lagt ut på Internett under en GPL-lisens. Disse ble publisert på Nagios Exchange [130] og Monitoring Exchange [131, 132], etter godkjenning fra oppdragsgiver. På begge sidene har vi selv funnet flere av plugin-ene benyttet i prosjektet, og vi håper at andre får nytte av vårt arbeid.

For plugin-en “xen_api” ble det rapportert tilbake til utviklerene at sjekker antagelig ikke blir utført i henhold til beskrivelsen (e-post ligger i I).

6.4 Konklusjon

Det viktigste målet med denne prosjektoppgaven ble oppnådd. En ny overvåkningsløsning ble satt i produksjon med overvåkning av store deler av IKT-avdelingens infrastruktur og servertjenester. Løsningen kjører nå parallelt med den overvåknigsløsningen som fantes fra før, og overvåker mye som den gamle løsningen ikke har mulighet til. Oppdragsgiver planlegger at den eksisterende overvåknigsløsningen skal fases ut over tid, etterhvert som resterende tjenester og enheter integreres i den nye løsningen.

Etter gruppens mening vil oppgaven kunne tjene som eksempel på de aller fleste tjenester og enheter ved utvidelse av løsningen. I forhold til Servers Alive er løsningen presentert i denne rapporten mer oversiktlig og tilrettelagt for utvidelse, noe som antagelig kan føre til mer effektiv drift for IKT-avdelingen.

Gruppen er fornøyd med resultatet som ble overlevert. Læringsutbyttet har vært høyt, og vi mener vi har nådd de målene vi satte ved starten av prosjektet. Ved overlevering var ikke alle IKT-avdelings servere og tjenester under overvåkning, men det er lite som skal til for å legge til nye enheter. Dersom det skal legges til nye sjekker vil det som er satt opp tjene som gode eksempler på hva som er mulig, og hvordan det kan settes opp.

Bibliografi

[1] Woodstone bvba, "Servers alive hjemmeside." http://www.woodstone.nu/salive/, 2013. [Online; accessed January-2013]. 1

[2] Wikipedia, "Iterative and incremental development — wikipedia, the free encyclopedia." http://en.wikipedia.org/w/index.php?title=Iterative_and_incremental_development&oldid=532022769, 2013. [Online; accessed 20-January-2013]. 6

[3] C. J. H. Thomas A. Limoncelli and S. R. Chalup, *The Practice of System and Network Administration*. Pearson Education, 2 ed., 2007. [Offline; accessed March-2013]. 10

[4] Teamquest, "Itil service support." http://www.teamquest.com/solutions/itil/service-support/, 2013. [Online; accessed May-2013]. 11

[5] ITIL Foundations, "Capacity management." http://www.itilfoundations.com/processes/capacity-management/, 2013. [Online; accessed May-2013]. 11

[6] L. Dragich. http://apmdigest.com/event-management-reactive-proactive-or-predictive, 2013. [Online; accessed March-2013]. 11

[7] Wikipedia, "Comparison of network monitoring systems." http://en.wikipedia.org/wiki/Comparison_of_network_monitoring_systems, 2013. [Online; accessed March-2013]. 12

[8] Zenoss Inc, "Zenoss customers." http://www.zenoss.com/customers/index, 2013. [Online; accessed January-2013]. 12

[9] Zenoss Inc, "Zen packs." http://wiki.zenoss.org/Category:ZenPacks, 2013. [Online; accessed January-2013]. 12

[10] Observium, "Observium main page." http://www.observium.org/wiki/Main_Page, 2013. [Online; accessed January-2013]. 12

[11] Zabbix SIA, "Zabbix." http://www.zabbix.com/index.php, 2013. [Online; accessed January-2013]. 12

[12] K. A. K. . N. Rusalan, "Comparison report on network monitoring systems." http://knowledge.oscc.org.my/practice-areas/rnd/benchmark-report/comparison-report-on-network-monitoring-system/at_download/file, 2010. [Online; accessed February-2013]. 12

[13] Wikipedia, "Nagios — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Nagios, 2013. [Online; accessed January-2013]. 12

[14] E. Hansen, "The monitoring setup part 1." http://www.linux.org/article/view/the-monitoring-setup-part-1-installing-centreon-nagios-ndoutils, 2012. [Online; accessed] March-2013. 12

[15] R. Gedda, "5 open source network management projects to watch." http://www.cio.com.au/article/373428/5_open_source_network_management_projects_watch/, 2011. [Online; accessed January-2013]. 12

[16] SecTools.org, "Top 125 network security tools." http://sectools.org/tag/traffic-monitors/, 2013. [Online; accessed January-2013]. 12

[17] Nagios Enterprises, "Nagios community." http://nagios.org/about/community, 2013. [Online; accessed January-2013]. 12

[18] Nagios Enterprises, "Nagios exchange." http://exchange.nagios.org, 2013. [Online; accessed January-2013]. 12

[19] NETWAYS GmbH, "Monitoring exchange." https://www.monitoringexchange.org/, 2013. [Online; accessed February-2013]. 12

[20] Icinga: Open Source Monitoring, "Icinga architecture." https://www.icinga.org/about/architecture/, 2013. [Online; accessed February-2013]. 13, 14

[21] Icinga Development Team, "The icinga-web rest api." http://docs.icinga.org/latest/en/icinga-web-api.html, 2013. [Online; accessed January-2013]. 13

[22] Icinga: Open Source Monitoring, "Icinga´s new & classic web interfaces." https://www.icinga.org/about/icingaweb/, 2013. [Online; accessed January-2013]. 13

[23] C. Haen*, E. Bonaccorsi, and N. Neufeld, "Distributed monitoring system based on icinga." http://accelconf.web.cern.ch/accelconf/icalepcs2011/papers/wepmu035.pdf, 2013. [Online; accessed March-2013]. 14

[24] Wikipedia, "Exit status — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Exit_code, 2013. [Online; accessed April-2013]. 18

[25] Nagios Plugin Development Team, "Nagios plugins." http://nagiosplugins.org, 2012. [Online; accessed January-2013]. 18

[26] Icinga Development Team, "Icinga plugin api." http://docs.icinga.org/1.8/en/pluginapi.html, 2013. [Online; accessed March-2013]. 18

[27] Icinga Development Team, "Standard macros in icinga." http://docs.icinga.org/latest/en/macrolist.html, 2013. [Online; accessed March-2013]. 20, 75

[28] Icinga Development Team, "Host and service dependencies." http://docs.icinga.org/latest/en/dependencies.html, 2013. [Online; accessed February-2013]. 25

[29] Icinga Development Team, "Integration with other software." http://docs.icinga.org/1.8/en/ch09.html, 2013. [Online; accessed] March-2013. 26

[30] Icinga Development Team, "Additional software." http://docs.icinga.org/1.8/en/ch10.html, 2013. [Online; accessed April-2013]. 26

[31] OpenSSH, "Openssh ssh client (remote login program)." http://www.openbsd.org/cgi-bin/man.cgi?query=ssh, 2013. [Online; accessed April-2013. 26

[32] D. Mauro and K. Schmidt, *Essential SNMP*. O'Reilly Media, 1 ed., 2001. [Offline; accessed March-2013]. 27

[33] The Internet Society, "Rfc3584." http://tools.ietf.org/html/rfc3584, 2003. [Online; accessed March-2013]. 27

[34] J. Davies, "Simple network management protocol." http://technet.microsoft.com/en-us/library/bb726987.aspx, 2006. [Online; accessed April-2013]. 28

[35] Net-SNMP, "Simple network management protocol." http://www.net-snmp.org/, 2013. [Online; accessed February-2013]. 28

[36] Microsoft, "Connecting to wmi remotely starting with windows vista." http://msdn.microsoft.com/en-us/library/windows/desktop/aa822854%28v=vs.85%29.aspx, 2012. [Online; accessed May-2013]. 28

[37] Icinga Development Team, "Service and host check scheduling." http://docs.icinga.org/latest/en/checkscheduling.html, 2013. [Online; accessed March-2013]. 31

[38] APC, "Netbotz 200." http://www.apc.com/resource/include/techspec_index.cfm?base_sku=NBRK0201&tab=software, 2013. [Online; accessed March-2013]. 34

[39] Icinga: Open Source Monitoring, "Icinga in action." https://www.icinga.org/users/, 2013. [Online; accessed February-2013]. 35

[40] Debian Monitoring Project, "Introduction." http://debmon.org/, 2013. [Online; accessed March-2013]. 35

[41] MonKey, "Select servicegroup from searchresult in icinga-web throws error." https://dev.icinga.org/issues/3692, 2013. [Online; accessed February-2013. 35

[42] J. G. A. H. Max Schubert, Derrick Bennett and J. Strand, *Nagios 3 Enterprise Network Monitoring: Including Plug-Ins and Hardware Devices*, ch. 2. Syngress, 1 ed., 2008. [Offline; accessed February-2013]. 36

[43] Standalone Sysadmin, "Nagios configuration howto." http://www.standalone-sysadmin.com/blog/2009/07/nagios-config/, 2009. [Online; accessed February-2013]. 36

[44] NConf, "Nconf - enterprise nagios configurator." http://www.nconf.org, 2013. [Online; accessed April-2013]. 36

[45] NagiosQL, "Web based administration tool designed for nagios." http://www.nagiosql.org, 2013. [Online; accessed February-2013]. 36

[46] Wikidot.com, "http://graphite.wikidot.com/." http://graphite.wikidot.com/, 2012. [Online; accessed February-2013]. 39

[47] J. Goldschrafe, "Metricinga." https://github.com/jgoldschrafe/metricinga, 2013. [Online; accessed March-2013]. 39

[48] NSClient++, "Nsclient." http://www.nsclient.org/nscp/, 2013. [Online; accessed February-2013]. 39

[49] NSClient++, "Version overview of nsclient." http://www.nsclient.org/nscp/downloads, 2013. [Online; accessed February-2013]. 40

[50] Icinga Development Team, "Monitoring windows machines." http://docs.icinga.org/latest/en/monitoring-windows.html, 2013. [Online; accessed January-2013]. 40

[51] R. Walker, "Examining load average." http://www.linuxjournal.com/article/9001, 2006. [Online; accessed April-2013]. 43

[52] Wikipedia, "Load (computing) — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Load_%28computing%29, 2013. [Online; accessed March-2013]. 43

[53] Wikipedia, "Paging performance — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Paging#Performance, 2013. [Online; accessed February-2013]. 44

[54] J. Hancock, "Check mem plugin." https://raw.github.com/jasonhancock/nagios-memory/master/plugins/check_mem, 2012. [Online; accessed March-2013]. 44

[55] Siolor, "Server 2003 port 3389 not listening after applying sp2." http://www.petri.co.il/forums/showthread.php?t=41357, 2009. [Online; accessed April-2013]. 45

[56] Wikipedia, "Lightweight directory access protocol — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol, 2013. [Online; accessed April-2013]. 46

[57] Wikipedia, "Domain name system — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Domain_Name_System, 2013. [Online; accessed April-2013]. 46

[58] Wikipedia, "Dynamic host configuration protocol — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/DHCP, 2013. [Online; accessed April-2013]. 46

[59] Nagios Plugin Development Team, "Check http." http://nagiosplugins.org/man/check_http, 2012. [Online; accessed February-2013]. 48

[60] SolarWinds Worldwide, "Exchange server monitoring best practices." http://www.solarwinds.com/resources/videos/exchange-server-monitoring-best-practices.html, 2012. [Online; accessed April-2013]. 49, 50

[61] echnet, "2010." http://gallery.technet.microsoft.com/office/Performance-and-Threshold-d32ff5a6, 2013. [Online; accessed March-2013]. 50

[62] M. Flacke, "Check multi." http://my-plugin.de/wiki/projects/check_multi/start, 2011. [Online; accessed March-2013]. 50

[63] Nagios Plugin Development Team, "Check cluster." http://nagiosplugins.org/man/check_cluster, 2013. [Online; accessed March-2013. 50

[64] Wikipedia, "Comparison of relational database management systems — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Comparison_of_relational_database_management_systems, 2013. [Online; accessed April-2013]. 51

[65] R. Schrag, "Tuning oracle's buffer cache." http://www.dbspecialists.com/files/presentations/buffercache.html, 2013. [Online; accessed March-2013]. 52

[66] G. A. Larsen, "Top 10 sql server counters for monitoring sql server performance." http://www.databasejournal.com/features/mssql/article.php/3932406/Top-10-SQL-Server-Counters-for-Monitoring-SQL-Server-Performance.htm, 2011. [Online; accessed February-2013]. 52

[67] Oracle, "Oracle databaseclient linux." http://www.oracle.com/technetwork/topics/linuxx86-64soft-092277.html, 2013. [Online; accessed March-2013]. 52

[68] Debian wiki, "Alien." http://wiki.debian.org/Alien, 2009. [Online; accessed March-2013]. 52

[69] The Oracle FAQ, "Database concepts and architecture." http://www.orafaq.com/wiki/Database_Concepts_and_Architecture, 2013. [Online; accessed January-2013]. 53

[70] B. Bruns, "Freetds." http://www.freetds.org/, 2011. [Online; accessed April-2013]. 54

[71] C. C. . S. S. GmbH, "Consol open source monitoring." http://www.consol.com/open-source-monitoring/database-monitoring/, 2013. [Online; accessed April-2013]. 54

[72] Oracle, "Privileges provided by mysql." http://dev.mysql.com/doc/refman/5.0/en/privileges-provided.html#priv_usage:, 2013. [Online; accessed February-2013]. 54

[73] Oracle, "Mysql cluster overview." http://dev.mysql.com/doc/refman/5.0/en/mysql-cluster-overview.html, 2013. [Online; accessed May-2013]. 55

[74] Oracle, "Download mysql cluster." dev.mysql.com/downloads/cluster, 2013. [Online; accessed April-2013]. 55

[75] S. W. Carter, "Ndb node monitoring." https://www.monitoringexchange.org/inventory/Check-Plugins/Database/MySQL/NDB-node-monitoring, 2009. [Online; accessed March-2013]. 55

[76] Cisco Systems, "Cisco snmp object navigator - ciscoenvmontemperaturestate." http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en&translate=Translate&objectInput=1.3.6.1.4.1.9.9.13.1.3.1.6, 2004. [Online; accessed May-2013]. 57

[77] G. Lausser, "Check network component health." http://labs.consol.de/nagios/check_nwc_health/, 2012. [Online; accessed April-2013]. 58

[78] raprop, "Check powerconnect switch." http://exchange.nagios.org/directory/Plugins/Hardware/Network-Gear/Dell/Check-PowerConnect-Switch/details, 2012. [Online; accessed April-2013]. 58

[79] Cisco Systems, "Best practices for monitoring cisco unified border element with cisco prime collaboration." http://www.cisco.com/en/US/prod/collateral/netmgtsw/ps6504/ps6528/ps12363/white_paper_c11-726161.html, 2013. [Online; accessed April-2013]. 58

[80] Cisco Systems, "Troubleshooting high cpu utilization on cisco routers." http://www.cisco.com/en/US/products/hw/routers/ps133/products_tech_note09186a00800a70f2.shtml, 2013. [Online; accessed April-2013]. 58

[81] Cisco Systems, "Asa 8.3 and later: Monitor and troubleshoot performance issues." http://www.cisco.com/en/US/products/ps6120/products_tech_note09186a0080b8e100.shtml#showmemory, 2013. [Online; accessed April-2013]. 58

[82] Cisco Systems, "Troubleshooting memory problems." http://www.cisco.com/en/US/products/sw/iosswrel/ps1831/products_tech_note09186a00800a6f3a.shtml, 2013. [Online; accessed April-2013]. 58

[83] C. Systems, "Configuring failover." http://www.cisco.com/en/US/docs/security/asa/asa80/configuration/guide/failover.html#wp1058096, 2013. [Online; accessed April-2013]. 59

[84] P. Sourdeau, "Cisco - check firewall asa and pix." http://exchange.nagios.org/directory/Plugins/Hardware/Network-Gear/Cisco/Cisco--2D-Check-firewall-ASA-and-PIX/details, 2008. [Online; accessed April-2013]. 59

[85] M. P. Gómez, "cisco asa sessions." http://exchange.nagios.org/directory/Plugins/Uncategorized/Software/SNMP/cisco_asa_sessions/details, 2008. [Online; accessed March-2013]. 60

[86] Cisco Systems, "CISCO-FIREWALL-MIB." http://tools.cisco.com/Support/SNMP/do/BrowseMIB.do?local=en&step=2&mibName=CISCO-FIREWALL-MIB, 2005. [Online; accessed April-2013]. 60

[87] Cisco Systems, "Introduction to cisco ios netflow - a technical overview." http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html, 2013. [Online; accessed April-2013]. 60

[88] Cisco Systems, "Netflow performance analysis." http://www.cisco.com/en/US/technologies/tk543/tk812/technologies_white_paper0900aecd802a0eb9.html, 2013. [Online; accessed April-2013]. 60

[89] Cisco Systems, "How to calculate bandwidth utilization using snmp." http://www.cisco.com/en/US/tech/tk648/tk362/technologies_tech_note09186a008009496e.shtml, 2013. [Online; accessed April-2013]. 60

[90] Cisco Systems, "Snmp counters: Frequently asked questions." http://www.cisco.com/en/US/tech/tk648/tk362/technologies_q_and_a_item09186a00800b69ac.shtml, 2013. [Online; accessed April-2013]. 60

[91] G. J. Frater, "Check iftraffic64." http://exchange.nagios.org/directory/Plugins/Network-Connections-Stats-and-Bandwidth/check_iftraffic64/details, 2009. [Online; accessed April-2013]. 60

[92] op5 AB, "op5." http://www.op5.com/, 2013. [Online; accessed January-2013]. 61

[93] VMware Inc, "vsphere sdk for perl documentation." http://www.vmware.com/support/developer/viperltoolkit/, 2013. [Online; accessed February-2013]. 61

[94] Wikipedia, "Simple object access protocol." http://en.wikipedia.org/wiki/SOAP, 2013. [Online; accessed March-2013]. 61

[95] VMware, "Performance intervals." http://pubs.vmware.com/vsphere-51/index.jsp?topic=%2Fcom.vmware.wssdk.pg.doc%2FPG_Performance.18.6.html, 2007. [Online; accessed April-2013. 62

[96] VMware Inc, "vsphere monitoring and performance." http://pubs.vmware.com/vsphere-51/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-51-monitoring-performance-guide.pdf, 2013. [Online; accessed March-2013]. 62

[97] VMware Inc, "Cpu counters." http://www.vmware.com/support/developer/vc-sdk/visdk400pubs/ReferenceGuide/cpu_counters.html, 2009. [Online; accessed April-2013]. 62

[98] VMware Inc, "Memory counters." http://www.vmware.com/support/developer/vc-sdk/visdk400pubs/ReferenceGuide/memory_counters.html, 2009. [Online; accessed April-2013]. 63

[99] VMware Inc, "Data collection levels." http://pubs.vmware.com/vsphere-51/index.jsp#com.vmware.vsphere.vcenterhost.doc/GUID-25800DE4-68E5-41CC-82D9-8811E27924BC.html, 2007. [Online; accessed April-2013]. 63

[100] Eric Siebert, "Storage i/o bottlenecks in a virtual environment." http://solarwinds-marketing.s3.amazonaws.com/solarwinds/whitepapers/VM_Whitepaper_Bottlenecks.pdf, 2011. [Online; accessed May-2013]. 63

[101] Joseph Dieckhans, "Troubleshooting storage performance in vsphere – part 1 – the basics." http://blogs.vmware.com/vsphere/2012/05/troubleshooting-storage-performance-in-vsphere-part-1-the-basics.html, 2012. [Online; accessed May-2013]. 63

[102] VMware Inc, "Using esxtop to identify storage performance issues for esx / esxi." http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1008205, 2013. [Online; accessed May-2013]. 63

[103] XEN Wiki, "Xapi rrds." http://wiki.xen.org/wiki/XAPI_RRDs, 2013. [Online; accessed March-2013]. 64

[104] R. Dobson, "Using xenserver rrds." http://community.citrix.com/display/xs/Using+XenServer+RRDs, 2011. [Online; accessed April-2013. 64

[105] Net-SNMP, "Snmptrapd." http://www.net-snmp.org/docs/man/snmptrapd.html, 2004. [Online; accessed March-2013]. 66

[106] A. Burger, "Snmp trap translator." http://snmptt.sourceforge.net/, 2012. [Online; accessed March-2013]. 66

[107] A. Burger, "Snmp trap translator v1.3." http://snmptt.sourceforge.net/docs/snmptt.shtml#Variable-substitutions, 2010. [Online; accessed March-2013]. 67

[108] Wikipedia, "Luftfuktighet — wikipedia, the free encyclopedia." http://no.wikipedia.org/wiki/Luftfuktighet, 2013. [Online; accessed March-2013]. 68

[109] SUN Microsystems, "Sun microsystems data center site planning guide." http://docs.oracle.com/cd/E19065-01/servers.e25k/805-5863-13/ch3.html#98939, 2002. [Online; accessed April-2013]. 68

[110] A. S. of Heating, Refrigerating, and I. Air-Conditioning Engineers, "Environmental guidelines for datacom equipment." http://tc99.ashraetcs.org/documents/ASHRAE_Extended_Environmental_Envelope_Final_Aug_1_2008.pdf, 2008. [Online; accessed April-2013]. 68

[111] Cisco Systems, "Data center power and cooling." http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns944/white_paper_c11-680202.pdf, 2011. [Online; accessed April-2013]. 68

[112] N. El-Sayed, I. A. Stefanovici, A. A. H. George Amvrosiadis, and B. Schroeder, "Temperature management in data centers: why some (might) like it hot." http://dl.acm.org/citation.cfm?id=2254778, 2012. [Online; accessed March-2013]. 69

[113] M. Patterson, "The effect of data center temperature on energy efficiency." http://ieeexplore.ieee.org/xpl/login.jsp?reload=true&tp=&arnumber=4544393&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4544393, 2008. [Online; accessed April-2013]. 69

[114] T. Azran, "Check netbotz." http://exchange.nagios.org/directory/Plugins/Hardware/UPS/APC/check_netbotz/details, 2013. [Online; accessed March-2013]. 69

[115] D. Josephsen, *Building a Monitoring Infrastructure with Nagios*, ch. 1. Pearson Education, 1 ed., 2007. [Offline; accessed January-2013]. 70

[116] Icinga Development Team, "Detection and handling of state flapping." http://docs.icinga.org/1.8/en/flapping.html, 2012. [Online; accessed March-2013]. 71

[117] Perl.org, "Perl ldap." http://ldap.perl.org/, 2012. [Online; accessed February-2013]. 71

[118] Wikipedia, "Patch tuesday — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Patch_tuesday, 2013. [Online; accessed March-2013]. 73

[119] Wikipedia, "Sendmail — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Sendmail, 2013. [Online; accessed March-2013]. 75

[120] L. Denness, "Nagdash." https://github.com/lozzd/Nagdash, 2013. [Online; accessed March-2013]. 77

[121] M. Smith, "Nagios-api." https://github.com/xb95/nagios-api, 2013. [Online; accessed March-2013]. 77

[122] B. Erk, "Icinga-web rest api." https://wiki.icinga.org/display/Dev/Icinga-Web+REST+API, 2012. [Online; accessed March-2013]. 77

[123] Kzoon, "Ie render bug for label with large fontsize." https://github.com/highslide-software/highcharts.com/issues/1567, 2013. [Online; accessed April-2013]. 82

[124] A. Afridi, "jquery marquee." http://plugins.jquery.com/marquee/, 2013. [Online; accessed March-2013]. 83

[125] M. Friedrich, "Merge branch 'test/cgis' into next." https://github.com/dnsmichi/icinga/blob/master/cgi/outages.c, 2013. [Online; accessed March-2013]. 83

[126] The PHP Group, "usort." http://php.net/manual/en/function.usort.php, 2013. [Online; accessed March-2013]. 84

[127] K. Lyng, "Nrpe now has the option for encrypting network traffic using ssl/tls from openssl." https://github.com/KristianLyng/nrpe/blob/master/README.SSL, 2011. [Online; accessed March-2013. 87

[128] Icinga: Open Source Monitoring, "Icinga 2 redesigns open source monitoring." https://www.icinga.org/about/icinga2/, 2013. [Online; accessed May-2013]. 98

[129] Wikipedia, "Compatibility layer — wikipedia, the free encyclopedia." http://en.wikipedia.org/wiki/Compatibility_layer, 2013. [Online; accessed May-2013]. 98

[130] MonKey, "Check ndb node memory – nagios exchange." http://exchange.nagios.org/directory/Owner/monkey/1, 2013. [Online; accessed May-2013]. 98

[131] MonKey, "Check ndb node memory – monitoring exchange." https://www.monitoringexchange.org/inventory/Check-Plugins/Database/MySQL/Check-NDB-Node-Memory, 2013. [Online; accessed May-2013]. 98

[132] MonKey, "Sync contacts from active directory – monitoring exchange." https://www.monitoringexchange.org/inventory/Utilities/Misc/Sync-contacts-from-Active-Directory, 2013. [Online; accessed May-2013]. 98

Vedlegg

Vedlegg A

Arbeidslogg

Arbeidslogg for Gruppen

Av	Tid	Notat
Hele gruppen	09.11.2012	Møte med IKT avdelingen på fylkeshuset
Hele gruppen	04.01.2013	Første møte med Erik H.
Hele gruppen	12.01.2013	Sett på diverse løsninger for monitorerings programvare. Startet google docs dokument for forprosjekt. Forbredt møter mandag og tirsdag. Arbeidstid: 12:00 - 16:00
Hele gruppen	13.01.2013	Sett på diverse videoer og presentasjoner om monitorerings programvare. Arbeidstid 10:00 - 13:00
UKE 1-2		Totalt: 7 timer
Hele gruppen	14.01.2013	Møte med Erik H. Satt opp skisse til gant skjema til forprosjekt. Laget presentasjon til møte. Arbeidstid 09:00 - 17:30 Nils: 10:00 - 17:30
Hele gruppen	15.01.2013	Møte med Erik H, og Svein-Inge anngående forprosjekt og oppstart Jobbet med forprosjektet Arbeidstid: 08:00 - 17:00
Hele gruppen	16.01.2013	Diskusjoner og avklaringer rundt forprosjekt innad i gruppen. Fant struktur og skrev på forprosjekt Arbeidstid: Bjørn-Erik 08:00 - 16:00 Øyvind 08:00 - 10:00 12-13:00 14:00 - 16:00 Nils: 10:00 - 16:00
Hele gruppen	17.01.2013	Jobbet med forprosjekt. Sendte google-docs invitasjoner til forprosjektet sendt ut til oppdragsgiver og veileder for å få kommentarerer. Arbeidstid: 08:00 - 17:00
Hele gruppen	18.01.2013	Fått rettet diverse innspill fra oppdragsgiver i forprosjektet. Satt opp spørsmål til ErikH før mandagsmøtet. Avklart rundt lab-oppsett og jobbing på fylkeshuset tirsdag. Arbeidstid: 08:00-17:00
UKE 3		Totalt: Bjørn-Erik: 43.5 Nils: 40.5

		Øyvind: 40.5
Hele gruppen	21.01.2013	Møte med ErikH, tatt opp spørsmål anngående forprosjekt. Jobbet med maler rundt LaTeX Sette opp virtuellt labmiljø, testet funksjoner i icinga, nconf Arbeidstid Øyvind og Bjørn-Erik: 09 - 18:00 Nils: 10 - 19:00
Hele gruppen	22.01.2013	Oppstart på fylkeshuset, kort møte med oppdragsgiver Satt opp og startet konfigurering av labmiljø. Arbeidstid (på fylkeshuset): 09 - 15:45
Hele gruppen	23.01.2013	Jobbet med legg til og fjerne host i Icinga, + research rundt icinga. Prøvd å finne relevante papers. Portert google docs over i LaTeX. Arbeidstid: 10-17
Hele gruppen	24.01.2013	Jobbet med finish rundt forprosjekt og LaTeX. Lagt til diverse host i Icinga, fått overvåkning på diverse tjenester, gjort research rundt nagios config struktur Arbeidstid: 09 - 20
Hele gruppen	25.01.2013	Forberedt fagstoff rundt presentasjon tirsdag. Lest gjennom forprosjekt, gjorde noen endringer. Strukturert config i icinga. Alle: Arbeidstid: 09 - 16:30 Bjørn-Erik: 16:30 - 19:30
UKE 4		Totalt: Øyvind: 41.25 Nils: 41.25 Bjørn-Erik: 44.25
Hele gruppen	28.01.2013	Møte med ErikH. Ferdigstilt forprosjekt, levert i fronter. Sett på rapporteringsmodul. Forberedt til presentasjon tirsdag. Arbeidstid: Øyvind og Nils 10 - 16:30. Bjørn-Erik: 10 - 15:30 Alle: 20:00 - 23:00
Hele gruppen	29.01.2013	Presentasjon av Icinga på fykeshuset. Første iterasjonsmøte. Jobbet videre med oppsett av overvåkning av servere og nettverksutstyr. Arbeidstid (på fylkeshuset): 09 - 15:45
Hele gruppen	30.01.2013	Satt opp basis mal for rapport. Skrev på rapporten.

		Arbeidstid: Øyvind og Bjørn-Erik: 10 - 14:30 Nils: 10:00 - 15:30
Hele gruppen	31.01.2013	Skrev rapport. Arbeidstid: 09 - 12:00. 14-15
Hele gruppen	1.02.2013	Andre fag
UKE 5		Totalt Øyvind 25.75 Nils 26.75 Bjørn-Erik 23.75
Hele gruppen	04.02.2013	Så på dependencies. Leste teori. Konfigurerte test for sammenligning av trafikk ved agenter. Så på liste over IKT-avds tjenester. Arbeidstid: 10:00 - 18:30
Hele gruppen	05.02.2013	Fått en uformel gjennomgang av hvordan HFK ser for seg å overvåke applikasjoner Sett på service dependencies og servicegroups Arbeidstid (på fylkeshuset): 09:00 - 16:00
Hele gruppen	06.02.2013	Satt opp diverse servicegroups i lab, sett at dette løser noe av applikasjonsovervåkningen. Sett på hvordan måle skalering. Arbeidstid: Bjørn-Erik: 10:00 - 20:00 Nils: 10:00 - 18:00 Øyvind: 13 - 20:00
Hele gruppen	07.02.2013	Satt opp ldap autentisering mot icinga-web og icinga-classic. Satt opp icinga-mobile. Kjørt tester på CPU tid osv, analyseres videre. Arbeidstid: 08:30 - 17:30
Hele gruppen	08.02.2013	Ryddet og jobbet med config på nsclient. WMI-querys - Nils Satt opp NRPE på Linux, ryddet config - Øyvind Jobbet med redundante oppsett - Bjørn-Erik Arbeidstid: 10:30 - 18:30
UKE 6		Totalt Bjørn-Erik 42.5 Nils 38.5 Øyvind 39.5
Hele gruppen	11.02.2013	Statusmøte med Erik

		Bearbeidet statistikk, kjørt ytelsestester. Testet check_multi Arbeidstid: 10:00 - 17:00
Hele gruppen	12.02.2013	Andre iterasjonsmøte Konfigurert klar brannmur for nettverks iterasjon. Satt opp produksjonsserve Arbeidstid (på fylkeshuset): 9:00 - 16:00
Hele gruppen	13.02.2013	SQL overvåkning, gjort klart for testing på exchange/mssql Arbeidstid: 10:00 - 17:00
Hele gruppen	14.02.2013	Noe restrukturering av konfig. Prøvd å fått in DB driver til MSSQL Arbeidstid: 9:00 - 17:00
Hele gruppen	15.02.2013	Strukturert SQL konfig Aribeidstid: 8:30 - 16:00
UKE 7		TOTALT Alle: 35.5
Hele gruppen	18.02.2013	Statusmøte med Erik Hentet inn statistikk fra diverse sjekker Labbet på SQL oppsett Arbeidstid: 10:00 - 19:00
Hele gruppen	19.02.2013	Fått inn infrastruktur i produksjon, satt opp basissjekker Avtalt vidre utrulling til produksjon med Lasse Arbeidstid (på fylkeshuset): 10:00 - 16:00
Hele gruppen	20.02.2013	Arbeidstid: 11:00 - 1600
Hele gruppen	21.02.2013	Satt opp konfig for kantswitcher og fikk dette inn på produksjonsservveren. Sett på generering for fysisk infrastruktur i Icinga Prøvde å finne ut hvilke muligheter vi har for overvåkning av Dell-switchene og hvordan vi kan utvide check_nwe. Lagt til nye enheter for overvåkning, primært ping. Sett på innkjøp til servermiljø. Funnet mer ut av nettverksstrukturen på fylkeshuset. Arbeidstid (på fylkeshuset): 9:00 - 16:30
Hele gruppen - Bjørn-Erik	22.02.2013	Satt opp punkter for samtale med de som sitter på servicedesk. Lagde utkas til mail til andre fylkeskommuner.

		Arbeidstid: 9:30 - 15:00
UKE 8		TOTALT Alle 33
Hele gruppen	25.02.2013	Sett på cisco failover. Skrev om SNMP. Arbeidstid: 10:00 - 1700
Hele gruppen	26.02.2013	Satt opp tester for cisco failover, lagt til flere enheter produksjonsnett (UPS apc, Dell blade chassis, HP Procurve switch) Sett på muligheter for overvåkning av VMware og Xen Satt opp graphite og graphios. Så på hvordan en kan overvåke trådløstkontroller, UPS og bladechassis. Arbeidstid (på fylkeshuset): 9:00 - 16:30
Hele gruppen	27.02.2013	Generert statusrapporter, lagt disse tilgjengelig på nett Arbeidstid: 11:00 - 1700
Hele gruppen	28.02.2013	Så på meru-controller og LDAPS autentisering. Lagt inn flere UPSer i overvåkningen. Laget et mer generisk måte for å kjøre sjekker over snmp. Fått inn resterende typer UPSer Arbeidstid (på fylkeshuset): 9:00 - 16:30
Hele gruppen	01.03.2013	Forbredt presentasjon til tirsdag. Arbeidstid: 12:00 - 16:30
UKE 9		Totalt Alle 33.5
Hele gruppen	04.03.2013	Ingen veiledning med Erik Startet med modul varsling, sett på contacts og contacts group, lest litt om filter i icinga. Satt opp snmptrapd i lab. Arbeidstid 10:30 - 17:00
Hele gruppen	05.03.2013	Satt opp snmptrapd på icinga. Satt opp til å motta traps fra meru-controllere. Lagt til flere UPSer i overvåkning Satt opp varsling på epost og SMS.

		Arbeidstid på fylkeshuset 9:00 - 16:00
Hele gruppen	06.03.2013	Rapportskriving Arbeidstid 9:00 - 18:00
Hele gruppen	07.03.2013	Statusmøte med Lasse, skiftet inputmetode av data til Graphite. Definerte tjenester for passiv input over SNMP. Testet å lagre informasjon fr alle sjekker i database. Endret bestilling av serverromovervåkningsenhet. Lagt til flere brannmurer og wlan-kontrollere i overvåkningen. Konfigurerte ndb_mgm for å overvåke mysql-cluster. Arbeidstid på fylkeshuset 9:00 - 16:00
Hele Gruppen	08.03.2013	Skrevet rapport. Arbeidstid 10:00 - 16:00
UKE 10		TOTALT Alle: 35.5
Hele gruppen	11.03.2013	Statusmøte med Erik Skrev script for syncing av kontakter og kontaktgrupper i AD. Skrevet rapport Arbeidstid: 10 - 19:00
Hele gruppen	12.03.2013	Skrev ferdig sync-script. Arbeidstid: 09 - 16:00
Hele gruppen	13.03.2013	Lagt inn domenekontrollere med LDAP og DNS sjekk i prod Opprettet bruker og startet med å teste plugins for uthenting av informasjon fra Citrix XenServer og VMware vCenter Restrukturerert deler av konfig i forhold til varsling Servicevindu: Lagt inn NSclient på utvalgte TSer. Lagt inn NSclient på utvalgte domenekontrollere. Testet Cisco Failover oppførsel i prod. Satt opp og testet round-trip mail sjekk Arbeidstid på fylkeshuset: 10:00 - 21:00
Hele gruppen	14.03.2013	Satt opp sjekk for mysql-cluster NDB med ndb_adm. Fortsatte med å sette opp overvåkning av Xen og ESX clusters Generelt oppsett av konfig i forhold til varsling Arbeidstid på fylkeshuset: 9:30 - 16:00
Hele gruppen	15.03.2013	Leste beskrivelsen for ulike counters for cluster, hosts og vm’er hentet ut fra

		vCenter, ulike artikler rundt hva folk monitorerte innenfor virtualesering. Rapportskriving. Fant og leste relaterte bacheloroppgaver. Arbeidstid: 08:30 - 13:00
UKE 11		TOTALT Alle: 38
Øyvind	17.03.2013	Satte opp og testet timeperiod for servicevindu i lab. 2 timer
Hele gruppen	18.03.2013	Skrev om sync av kontakter. Arbeidstid: 09:00 - 16:00
Hele gruppen	19.03.2013	Skrev ndb-sjekk for minne i perl. Statusmøte/demo av Icinga. Møte med servicedesk-ansvarlige om GUI for statusvindu. Så på eksisterende dash og sql-struktur. Arbeidstid på fylket: 0900 - 15:45
Hele gruppen	20.03.2013	Skrev statusrapport for varsling-iterasjonen. Klargjort rapport for første gjennomlesning av Erik H. Arbeidstid: 09:00 - 20:00
	21.03.2013	Hjemmeeksamen i kvalitetsledelse.
Hele gruppen	22.03.2013	Så på Icinga-web api. Fant queries for å hente service og host-status og hvordan filtrere dette. Arbeidstid: 09:00 - 16:00
UKE 12		TOTALT Nils og Bjørn-Erik: 31.75, Øyvind: 33.75
Bjørn-Erik og Øyvind	25.03.2013	Satt opp nagdash og skrev om til å benytte data fra Icinga-web. Bjørn Erik: 10:00 - 14:00 Øyvind: 10:00 - 16:00
Hele gruppen	26.03.2013	Servermiljø-boksen ble stabil. Satt opp sjekker for temperatur og luftfuktighe La inn tid på statusvindu. Lagde “API” for temperatur og luftfuktighetsensore La inn på statusvindu via javascript. Bjørn-Erik og Øyvind Arbeidstid: 09:00 - 15:45 Sett på kommentarer fra Erik i rapporten og skrevet om noe. Nils

		Arbeidstid: 12:00 - 16:00
	27.03 - 03.04	Påske
UKE 13		TOTALT Bjørn Erik: 10.75, Nils: 4, Øyvind: 12.75
Bjørn-Erik og Øyvind	04.04.2013	Fylkeshuset Lagde data-tilgang mot footprints for å hente ut saksstatistikk til statusvindu. 09:15 - 15:45
Bjørn-Erik og Øyvind	05.04.2013	Satt opp statusvindu på en webserver i lab. La til mulighet for dummy-data. 10:00 - 14:00
UKE 14		TOTALT Bjørn-Erik: 10.5, Nils: 0, Øyvind: 10.5
Hele gruppen	08.04.2013	Møte med veileder. Fikk input om å lese på HCI. Leste om CSS metoder for få overlay på bilder. Arbeidstid: 09:45 - 15:00
Hele gruppen	09.04.2013	Fylkeshuset Så på Oracle-tilkobling i produksjon. Satt opp kontakt-sync i produksjon. Skrev om script for sjekk av serverrom-miljø til å bruke terskelverdier fra Icinga. Fortsatte på dashboard. La til flere footprints felt. Leste om uthenting av custom variabler for å prioritere det som er nede. 09:30 - 15:45
Hele gruppen	10.04.2013	Fortsatte å se på custom variabler. Leste Icinga kildekode for å finne hvorda network outages kalkuleres. Arbeidstid 09:00 - 15:45
Hele gruppen	11.04.2013	Fylkeshuset Fortsatte å jobbe med statusvindu. Så på hvordan prioritere hoster og tjenester. Uthenting av custom variabler. 08:45 - 15:45
Hele gruppen	12.04.2013	Så på ulike bibliotek for grafing i javascript. Satte opp grafer over saker gjennom Highcharts. Fikk opp grafisk visning av serverrom for temperatur.

		Arbeidstid: 09:00 - 17:00
UKE 15		TOTALT Alle: 33.05
Hele gruppen	15.04.2013	Gjort klar MSSQL og Oracle konfig til fylkeshuset på tirsdag, skrevet mer utfyllende om denne implementasjonen Arbeidstid: 10:00 - 17:00
Hele gruppen	16.04.2013	Satt opp flere instanser i Oracle, fått orden på domene auth mot MSSQL, sett på thresholds. Endret layout på statusvisningen til å være side ved side stedet for under hverandre. Arbeidstid på fylkeshuset: 09:00 - 16:00
Hele gruppen	17.04.2013	Skrevet om og kommentert generering av hosts script. Rettet småfeil i CSS og rettet utseendet på elementer. Arbeidstid: 10:00 - 16:00
Hele gruppen	18.04.2013	Statusmøte med Svein-Inge Sett på thresholds, og overlevering av prosjekt til fylkeshuset Arbeidstid på fylkeshuset: 09:00 - 16:00
Hele gruppen	19.04.2013	Rapportskriving Nils Arbeidstid 10:00 - 16:00 Øyvind Satt opp sensor-overlay til å tillate at antall sensorer endre dynamisk: 12:00 - 20:00
UKE 16		TOTALT Bjørn-Erik: 27, Nils: 33, Øyvind: 35
Hele gruppen	22.04.2013	Statusmøte med Erik Skrevet status rapport for iterasjon 5 Laget TODO dokument med oppgaver frem til levering. Skrevet parser for RSS-feed til statusvindu. Arbeidstid: 10:00 - 16:00
Hele gruppen	23.04.2013	Sett over config, reboot test av icinga prod server Tatt bilder av performance data til rapport

		Satt opp host og service escalation eksempler Rettet på småfeil i thresholds Arbeidstid på fylkeshuset: 09:00 - 16:00
Hele gruppen Øyvind og Bjørn-Erik Nils	24.04.2013	Skrevet Init scripts i forhold til restart Icinga kan nå laste environment variabler. Alt skal nå kjøre automatisk etter reboot. Arbeidstid på fylkeshuset: 09:00 - 16:00 Gjennomgått og kommentert all kode for statusvindu. Fikset noen småbugs. 19:00 - 00:30 Skrevet dokument for opplæring 20:00 - 22:30
Hele gruppen Øyvind	25.04.2013	Hatt opplæring for Lasse og Svein-Inge Ryddet i konfig, generics, services Skrevet om og samlet sjekker. Arbeidstid på fylkeshuset: 09:00-16:00 Satt opp feilmedlinger hvis statuskjermen ikke har kontakt med back-end. 18:00 - 21:00
Alle	26.04.2013	Oppdatert referater for statusmøter Lagt ut statusrapporter for modul 5 og 6 Oppdatert visio tekninger med nyeste “Config status” 11:30 - 19:00
UKE 17		TOTALT Bjørn-Erik: 34.5, Nils: 40, Øyvind 43
Nils og Øyvind	28.04.2013	Satt opp spørsmål før møte med Erik på mandag. 17:00 - 19:00
Nils og Øyvind	29.04.2013	Møte med Erik Funnet feil og mangler i rapport. Sett på kode blokker i LaTeX Prøvd å få tak i ISO standard for brukergrensesnitt 10:00 - 16:00
Alle	30.04.2013	Kommentert konfig. Ryddet på server. Satt opp sjekk av båndbredde på WAN-porter. Sett på rapportering av alarmer, og gevinst av dette

		Fylkeshuset 08:00 - 16:00 Skrevet rapport. 18:30 - 01:00
Alle	1.05.2013	Rapport Overlevert kartlegging av behov til Svein-Inge for kommentarer. 10:00 - 20:00
Alle	2.05.2013	Overlevert bachelor til fylkeshuset Arbeidstid på fylkeshuset 10:00 - 16:00 Rapportskriving 18:00 - 22:00
Alle	03.05.2013	Rapportskriving 10:00 - 22:00
Alle	04.05.2013	Rapportskriving 10:00 - 22:00
UKE 17		TOTALT Bjørn-Erik: 67.5, Nils: 75.5, Øyvind 75.5
Alle	04.05.2013	Rapportskriving 10:00 - 22:00

Vedlegg B

Møtereferater

I dette vedlegget ligger møtereferater fra prosjektperioden.

Møter med Oppdragsgiver

Møte 29-01.2013

Tilstede: Svein-Inge, Lasse, Øyvind, Nils, Bjørn-Erik

Presentert kjærneprogramvare fra første iterasjon.
Labbing vil bli hovedsakelig i labnett.
Lasse vil være med å kjøre ferdige iterasjoner i prod/labmiljø etterhvert som dette er ønskelig.
Klarsignal for å fortsette med lab på icinga med plugins(Hovedpunkt).
Snakket om neste iterasjon(Server).
fått tilgang til mer labutstyr(2 switcher, brannmur, router).
Åpnet for IKT-avdelingen å gi kommentarer til presentasjon, skal gå igjennom Lasse og Svein-Inge.

Hevet

Møte 12.02.2013

Tilstede: Lasse, Thomas, Bjørn-Erik, Nils, Øyvind

På Statusmøte om server overvåkning ble det diskutert
Gruppere serviser i service grupper (For å overvåke applikasjoner).
Service dependencies.
Overvåkning av Windows servere.
Overvåkning av Linux servere.
Litt om sikkerhet (Kryptering på sjekker som kjøres).
Litt om utrulling (Til servere som skal overvåkes).

Lasse og Svein-Inge er fornøyd med fremgangen og vi får klarsignal på å gå videre med infrastrukturmodulen

Hevet

Møte 07.03.2013

Tilstede: Lasse, Svein-Inge, Nils, Øyvind, Bjørn-Erik

Presenterte status for server og infrastrukturiterasjonene. Snakket litt om mulig overvåkning av postfix-servere. Diskuterte status for ordre av serverromovervåkningsutstyr. Endte opp med å sende bestilling til lokal leverandør. Snakket om muligheter for å være med på neste service-vindu, 13.03, for å installere agenter og teste overvåkning.
Det ble bestemt at Lasse skal sende Nils mail om flere enheter skal til overvåkning nå.

Presentert følgende tema for IKT avdelingen:
Infrastruktur
- Status kart
- Forhold mellom enheter (parent/child)
- Overvåkning av nettverks utstyr
- Overvåkning av redundante systemer

Server
- Hardware monitorering
- Tjenester/applikasjoner
- SQL Servere
- Exchange
- Generisk applikasjon som en samling av tjenester

Hevet

Møte 08.03.2013

Tilstede: Lasse, Svein-Inge, Thomas, Nils, Øyvind, Bjørn-Erik

-Vaktgruppe i AD
Det opprettes vaktgruppe i AD. Den skal inneholde brukere med ansvaret for oppfølging ved feil etter normal arbeidstid og frem til kl 2100 på hverdager. Dette kan da være én til flere brukere, som vil variere fra uke til uke. Varsling må kunne utføres pr. epost og sms.
-Rolleansvarlig
Overvåking av sentrale enheter skal også ha mulighet for å gi varsling pr. epost og/eller sms til rolleansvarlig. AD grupper for hvert enkelt fagområde opprettes. Denne varslingen skal være aktiv hele døgnet.
-Utvidet varsling
Overvåking skal varsle kritiske feil i sentrale systemer i tidsrommet 2100 – 0700 på hverdager og hele døgnet i helg. Dette vil også gjelde for en bestemt AD gruppe. Denne varslingen skal utføres på sms.

Hevet

Møte 08.03.2013

Tilstede: Lasse, Svein-Inge, Nils, Øyvind, Bjørn-Erik

Viser eksempel på varsling til epost og sms, satt opp eksempler på kritisk varsling.
Vist sync script mot ad
Satt opp eksempler på service escalation

Svein-Inge og Lasse er fornøyd og vi går videre til neste modul statusvisning og serverromovervåkning

Hevet

Møte 18.04.2013

Tilstede: Lasse, Svein-Inge, Nils, Øyvind, Bjørn-Erik

Informerte om status for statusvindu og servermiljø. Avtalte brukertesting av statusvinduet ved å sette opp på servicedesk.
Diskuterte hva som gjenstod for implementasjonen. Svein Inge skal sjekke ut om plugins vi har skrevet kan deles videre. Mulighet for å få inn driftsfeed i Statusvindu, Øyvind følger opp dette.

Hevet

Statusmøter med veileder

Møte 04.01.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Bedre å gjenbruke kode
Gjør oppgaven vanskelig nok
Se på bachelor til Hans Åge
Mandager 10:30 - veiledningsmøter. Ukentlig
Avtale et møte med fylket og ErikH.
Erik er i Oslo hver Tors.
Detaljert kravspec. i forprosjekt. Koble til ITIL.
Beskrive innføringsprosessen / realisering av løsning
Sjekk slashdot / reddit. Munin
Hvorfor overvåker de? Hvordan blir det brukt?
Skrive rapport jevnlig gjennom prosjektperioden, denne er viktig.

Hevet

Møte 14.01.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Pratet med Erik 10:30 - 10:45
Sett opp en plan utfra gj.snitt
600 timer hver
Gantt på projektor

Snakket med Jon Langseth. 11:00 - 11:45
Fikk demo av IT-tjenestens overvåkningsløsning Nagios

Hevet

Møte 15.01.13 med oppdragsgiver og veileder

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik, Svein-Inge

Enkelt å legge til hosts.
Rutiner.
Legge til host i gruppe via web.
Ønsker ikke lisenser.
Servermiljø: finne boks, argumentere for. Må se helhetlig ut (rackmonterbar).
Trådløse punkter.
Er alt OK, trenger ikke noe vises.
Styre når det skal varsles på tid.

Vises på stoskjermer på fellesareal og servicedesk.
SMS og epost.
Ikkt Printere.
Rundt 300 hosts
UPS
Utvikling vs. implementasjon av eksisterende løsninger
Symantec Backup
Sikkerhet er viktig.
IDP løsning er nærstående, kan muligens brukes
Arvekjeder
Ta en tur till fylkeskommunen. Sette oss inn i nettverket
Har mulighet til lab.
Møte på forhand ved testing i produksjonsmiljø. Vurdere konsekvenser.
Utstyr
Holde trenddata utenfor foreløpig.
Hyppigere, men ikke lange møter
Begrense rettigheter ved overvåkning. Fylket setter opp adgang
Evt. ukentlig/annehver uke møte. Uansett sende oppsummering etter hver sprint.
2 uker sprint med møte ved avslutning.
Vi kaller inn til møte med agenda på noen ord
Viktig at vi velger noe vi synes er spennede å jobbe med.
Må kunne skalere bildet
Ide: større bilde ved krise / kriseskjerm
Hvordan fange oppmerksomhet?
Definere grupper, hva de kan se. Skoler og andre avdelinger.
Mandager og tirsdager best for Svein-Inge.

Hevet

Møte med ErikH 21.01.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Gruppering - sammenheng og avhengigheter? Få fram at det er arbeidsstrukturen.
Referanser der det er mulig. Samle som bibtex
Spør Hilde om tex-mal for bacheloroppgave
Ikke bruk for tid underveis på formattering i latex.
Dag Langmyr, latex for nybegynnere
Avvik fra forprosjekt. Dokumenter avvikene.
Få fram i gantt at vi skriver rapporten underveis. Ser "klipp/lim" ut.
Vil noen moduler se anerledes ut?
WBS er greit å ha med.
Erstatte "ferdigstille" med "utvikle" en løsning.
"der hendelser raskt kan oppdages og eskaleres." mer fortellende en målbart.
Gruppere kravene i effektmål. Dele omfattende og oversiktlig. Lett å administrere.
Del i 4: omfattende, fleksibelt.
Hilden, Drismo: Gjøre "noe" som bidrar tilbake til icinga.

Hevet

Møte med ErikH 28.01.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Status, satt opp icinga, labmijø på fylkeshuset, fått vpn, satt opp overvåkning på windows og linuxserver, sett på Nconf configurere via gui, virker som dette er det beste men vi er ikke imponert.
Legge til ny host via gui.
Tar imot endringer via gui, utfører disse, kjøre test config kommando før endring
Sikkerhetsproblematikk i skriving av konfigfiler.
Sjekk smnp vs ssh vs service.
Sikkerhetsproblematikk i forhold til ssh bruker.
Starte å skrive på rapport, få ned det vi har gjort på papir asap.
Modul er greit, kan bruke modulgrupper.
Skrive i mål, i forhold til dagens servers alive løsning vil den være med omfattende, fleksibel osv
Kunne skreddersy en overåkningsløsning for en smb bedrift
Har ikke noe å si hvor bilder blir liggende
b for bottom, i latex.
Vektor grafikk i latex er yesbox(ps eller pdf)

Hevet

Møte med ErikH 04.02.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

statusrapporter skal leveres i fronter, hva er dette?
Skal leveres i fronter
Webinterface for å legge til host (HFK er usikker på om dette trengs) rapport, litt oppstartshjelp
1. Hvor teknisk burde den være, generelt hva burde den inneholde?

Som å skrive for medstudenter
2. Hvor mye vi må forsvare valg av programvare i ønsker å bruke

Referere til mange forskjellige artikkler
Velge på skjønn
3. Kravspec, usikker fordi krav ikke er fastspikret, og overordnet

Skriv en naturlig kravspec, basert på hva vi trenger.
Kan være ramse opp hver enkelt plugin.

Ressursbruk / trafikk, kjøretid, burde være med i rapporten.
Finne frem nagios oppgave fra 10 år siden.
Atop, for å overvåke trafikk.

Hevet

Møte 11.02.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Ta med CMDB i rapporten.
Skriv et kapittel om sikkerhet. DOS-angrep?
Lag en rutine for installasjon.
Styr unna ompakking

Til påske: lage utkast av rapporten. Struktur og mye av innholdet.
Arbeidslogg kan bli sjekket mot Gantt.

Hevet

Møte 18.02.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

-SNMP query vs Trap
  Erik mener det kan være verdt å ha med begge deler
  Definitivt et tema for rapporten
-Statistikk for NRPE vs SSH
  Hent tall for user/kernelmode fra /proc/pid/stat
  Sjekk barchart_demo.py matplotlib
-Hjemmesiden
  Send mail til Erik med adressen
  Bør ha med forprosjektet og statusrapporter
  Presentasjon av gruppen

Hevet

Møte 25.02.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Ringtopologi - skrive om i rapporten. Forankre vitenskapelig.
VMware og Xen. Ikke så lett å få inn på kartet.

Grafer - finn andre farger. Gult på hvit fungerer dårlig.
Trafikk for SSH med ControlMaster. Bare kommenter. Ikke bruk tid på å dele opp målinger.

Mail til andre fylkeskommuner.
-Ha en forventning til hva som svares. Vit hva dere er ute etter å finne ut. Ta også forventinger og avvik med i rapporten.
Endre forprosjektet litt for å utelate noen detaljer om fylkeskommunen.
Exchange: prøve å generalisere hva som bør overvåkes på epostserver.

Statusrapporter: skal vise status.

Hevet

Møte 11.03.2013

Tilstede: Nils Slåen, Øyvind Sigerstad, Bjørn Erik Strand, Erik Hjelmås

Delta på servicevindu
Meru SNMP
Versjonoppdatering
Utstyr servermiljø
Fokus på kjerne
Rapport - Beslutninger
Kommandoer - ok i rapport...lengre kode / framgangsmåte-> vedlegg
Skriv om utfordrende config
Tegn trestruktur - vis egendefinerte filer - hel side, ikke for komplisert
Mail / SMS - vis at dykk er beviste på problemet
Dele opp varsling - script for kontaktINJECT
AD source for kontakt - what do they want
switch på statisk eller ad config
Rapportutkast før påske
Avsluttende kapittel med fremgangsmåte

Hevet

Møte 08.04.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Servermiljø. Skriv litt om forskjellige oppfatninger om temperatur. Les artikkel fra feb "Temperature Management in Datacenters" usenix. 38 nr 1.
Ikke bli for mye operations. Noen cases kan være interessant for fremvisning. Ikke la det bli noen stor del av oppgaven. Bruk heller tid på å få overvåkningen best mulig.
Ta med referat som vedlegg, henvis i rapporten - diskusjon rundt overvåkningsskjerm med helpdesk-ansatte. Kan skrives om, anonymiseres og "sensureres" til en viss grad.
Vis fram at dere har gjort noe som andre medstudenter kanskje ikke kunne klart.
Forankre statusvisning i "usability"-litteratur. HCI.
Forslag: Vise temperatur på en figur.
Fargevalg: rødt - fare vs varme
Brukertesting. La brukerne prøve ut, få tilbakemelding.

Hevet

Møte 15.04.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Statusvisning - prioritet. Ikke bruk for mye tid på tid. Diskuter i rapporten.
Hva finnes av status i icinga.
Bidra tilbake til Icinga. Legge ut plugins og evt statusvisning. Ta med i rapporten.
Thresholds. Standard vs. sette selv. Finne referanser. En egen tabell for alle thresholds. Evt. eget kapittel. Kan en automatisk sette thresholds?

Thresholds -> noen skal gjøre noe.
Så profesjonelt som mulig.

Vurder hvordan vi skal ta tilbakemeldinger for statusvindu. Hvilke spørsmål skal stilles. Hva forventes? Hvilken betydning er det av?

Hva har dere forsøkt å få til. Deler brukerne oppfatningen.

Jakob Nilsen - 10 Usability Heuristics for User Interface Design.

Hevet

Møte med ErikH 22.04.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Viste siste statusvisning-iterasjon.
Hvordan evt legge ut plugins/kode
Overlevering. Hvordan ta med i rapporten?
Framføringsdatoer
    Invitasjoner

Hevet

Møte 29.04.13

Tilstede: ErikH, Nils, Øyvind

-Oppdatering på hva vi har gjort
-Legge ved opplæringsdokumentet
-Bilder fra Icinga, anngi under bilde, gjengitt med tillatelse.
  -Open source bilder, sjekk wikimedia.
-Eposter
  -Samle som et vedlegg evt.
  -Trenger ikke gjengi all kommunikasjon
-Ved vedlegg som referanse, personlig kommunikasjon med tidspunkt
-Statusvindu, innleding med undersøkelse og data, så skrive hva.
  -Referater som vedlegg
-Selvlagde illustrasjoner, verktøybruk, Visio. Consolas - pen font. Deja Vu Sans Mono.
-Grafer fra graphios
  -Greit å bruke.
  -Innverter grafen kanskje.
  -Fint å bruke på presentasjonen.
-ISO, spør biblioteket eller Eivind Johansen.
-Fargeblindhet-test, ikke dra generelle konklusjoner.
-Uttrykk, skriv om begresbruk. Konsikvent bruk f.eks host og service. Kan evt. si at når vi snakker om service snakker vi om objektet. Ingen krise. Konsikvenshet.
-Noe man har skrevet selv før. Ingen problem med samme tekst. Ta evt. med som bakgrunn.
-Overlevering
  -Bevist på hva som skjer nå. Ikke gi supportgarantier, men kan være behjelpelig med spørsmål.

Hevet

Møte 06.05.13

Tilstede: ErikH, Nils, Øyvind, Bjørn-Erik

Rapport:
  mellompunkt mellom teori og utviklingsoppgave
- Erik mener dette er en utviklingsoppgave, med fokus på realisering.

  Use case
- Erik vet ikke om det skal brukes use case, han ser på dette til neste gang.
- Ferdig programvare som tilpasses og realiseres.
  Bruke diff som vedlegg
- Ta med diff, og forklar hva endringen som er gjort.
  Kilder
- Formulere setninger slik at kilderhenvisninger blir henvist
- Ikke noe problem å henvise til samme kilde flere ganger.
  Hvor skal grafing inn
- Tilbakemelding etter finlesning.
  Hostsjekk Icinga dokumentasjon er ikke riktig.
- Viktig at feil vi finner blir kommunisert tilbake til Icinga.
  Vi har ikke nok tid
- Skrive at vi har kommet dit vi skal komme for å dra dette videre.
- IKT-avdelingen skal klare å bruke dette videre.
- Er ferdig utviklet, men kan utvikles videre.
  Konklusjon
- Gå over målene som er satt.

Hevet

Vedlegg C

Presentasjoner

I dette vedlegget ligger presentasjoner som har blitt holdt for IKT avdelingen gjennom prosjektet.

Diverse løsninger

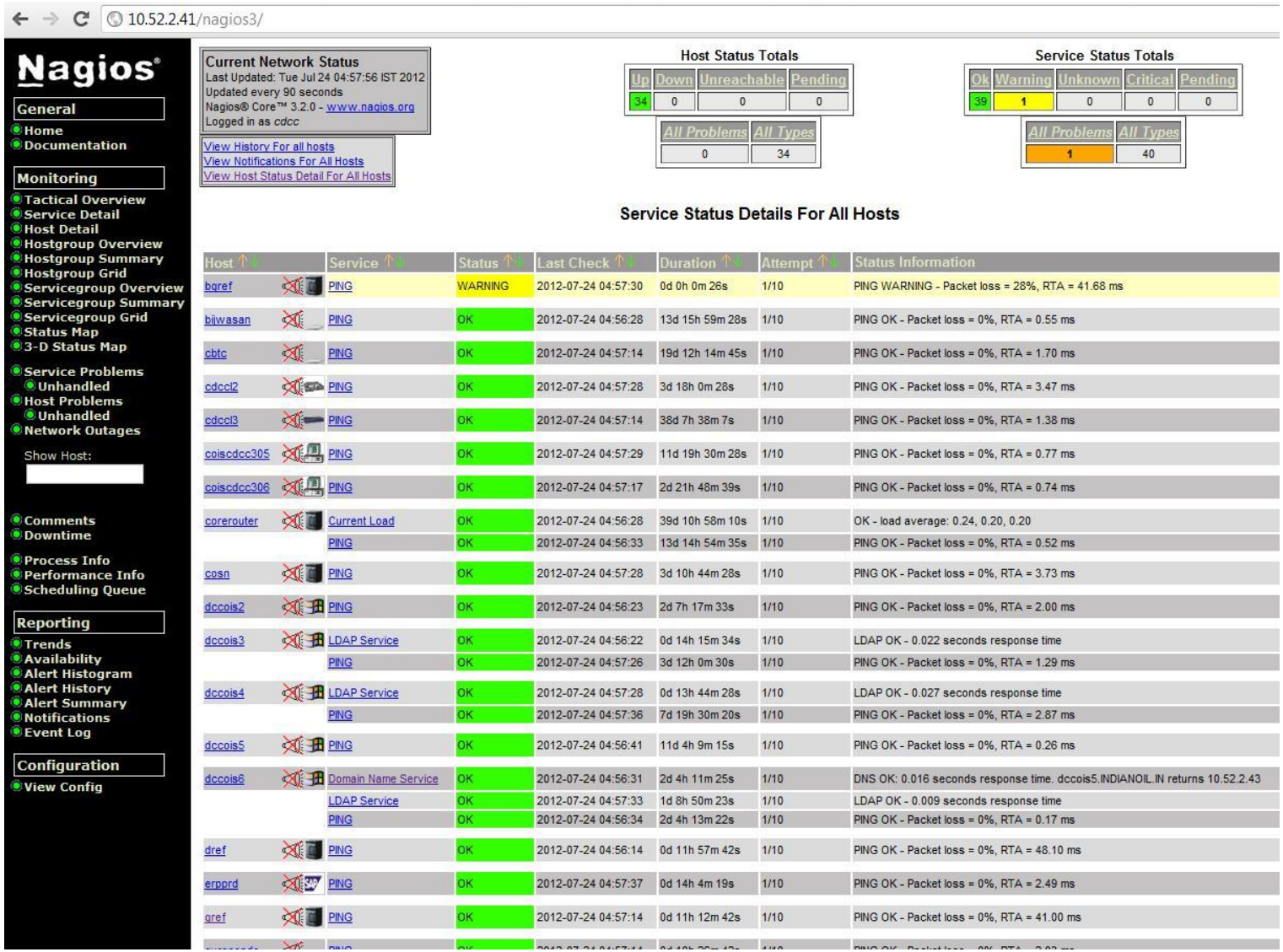
MonKey Presentasjon 15.01.2013

Monitoring is Key

Pakkeløsninger

- Fokus på "Up/Down"
- Enkelt web grensesnitt
- Rapport
- Konfigurasjon
- Krav
- SLA

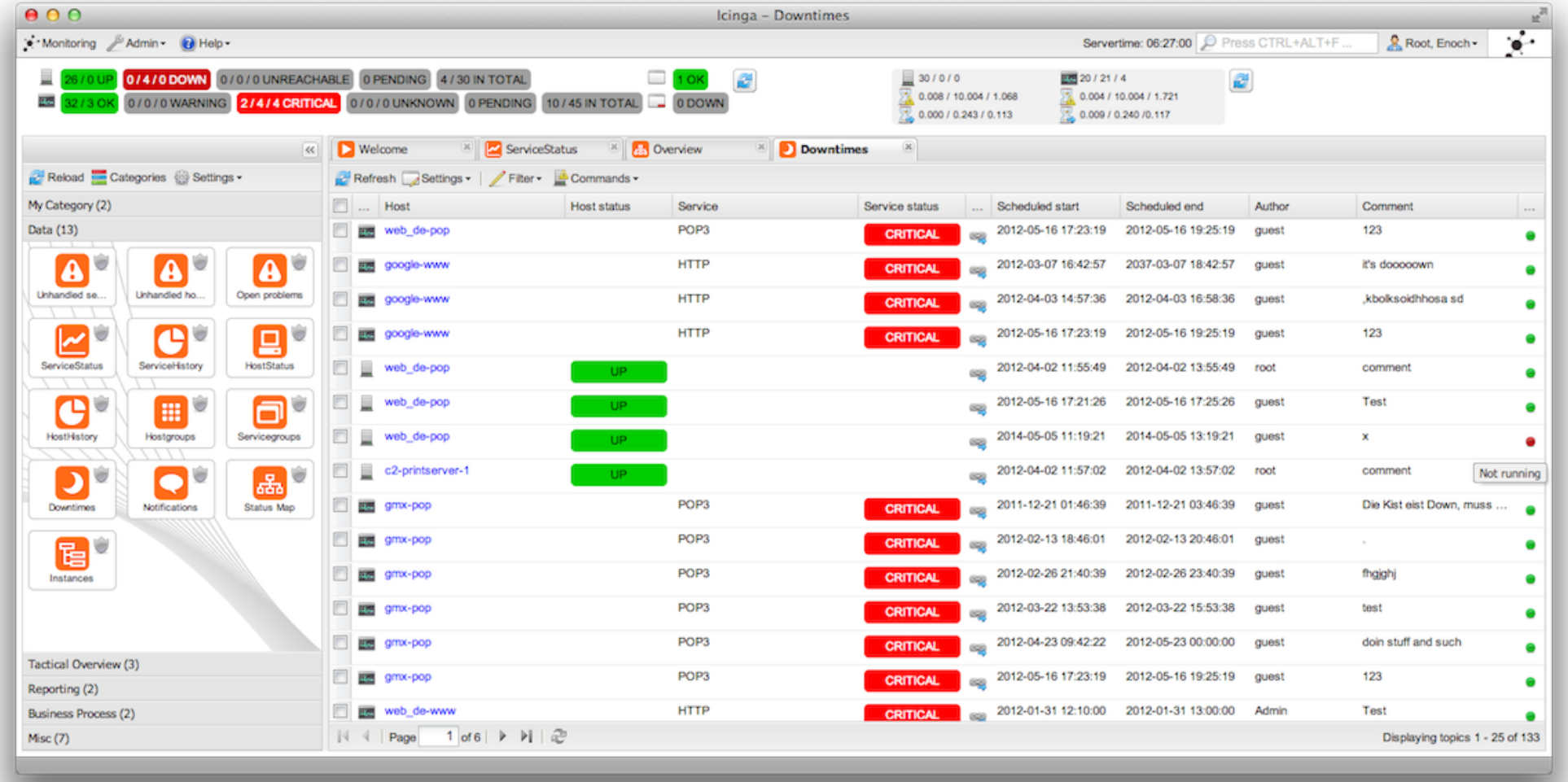
Nagios


10.52.2.41/nagios3/
Nagios
General
Home
Documentation
Monitoring
Tactical Overview
Service Detail
Host Detail
Hostgroup Overview
Hostgroup Summary
Hostgroup Grid
Servicegroup Overview
Servicegroup Summary
Servicegroup Grid
Status Map
3-D Status Map
Service Problems
Unhandled
Host Problems
Unhandled
Network Outages
Show Host:
Comments
Downtime
Process Info
Performance Info
Scheduling Queue
Reporting
Trends
Availability
Alert Histogram
Alert History
Alert Summary
Notifications
Event Log
Configuration
View Config
Current Network Status
Last Updated: Tue Jul 24 04:57:56 IST 2012
Updated every 90 seconds
Nagios® Core™ 3.2.0 - www.nagios.org
Logged in as cdcc
View History For all hosts
View Notifications For All Hosts
View Host Status Detail For All Hosts
Host Status Totals
Up 34
Down 0
Unreachable 0
Pending 0
All Problems 0
All Types 34
Service Status Totals
Ok 39
Warning 1
Unknown 0
Critical 0
Pending 0
All Problems 1
All Types 40
Service Status Details For All Hosts
Host
Service
Status
Last Check
Duration
Attempt
Status Information
bgref PING WARNING 2012-07-24 04:57:30 0d 0h 0m 26s 1/10 PING WARNING - Packet loss = 28%, RTA = 41.68 ms
biiwasan PING OK 2012-07-24 04:56:28 13d 15h 59m 28s 1/10 PING OK - Packet loss = 0%, RTA = 0.55 ms
cbtc PING OK 2012-07-24 04:57:14 19d 12h 14m 45s 1/10 PING OK - Packet loss = 0%, RTA = 1.70 ms
cdccl2 PING OK 2012-07-24 04:57:28 3d 18h 0m 28s 1/10 PING OK - Packet loss = 0%, RTA = 3.47 ms
cdccl3 PING OK 2012-07-24 04:57:14 38d 7h 38m 7s 1/10 PING OK - Packet loss = 0%, RTA = 1.38 ms
coiscdcc305 PING OK 2012-07-24 04:57:29 11d 19h 30m 28s 1/10 PING OK - Packet loss = 0%, RTA = 0.77 ms
coiscdcc306 PING OK 2012-07-24 04:57:17 2d 21h 48m 39s 1/10 PING OK - Packet loss = 0%, RTA = 0.74 ms
corerouter Current Load OK 2012-07-24 04:56:28 39d 10h 58m 10s 1/10 OK - load average: 0.24, 0.20, 0.20
PING OK 2012-07-24 04:56:33 13d 14h 54m 35s 1/10 PING OK - Packet loss = 0%, RTA = 0.52 ms
cosn PING OK 2012-07-24 04:57:28 3d 10h 44m 28s 1/10 PING OK - Packet loss = 0%, RTA = 3.73 ms
dccois2 PING OK 2012-07-24 04:56:23 2d 7h 17m 33s 1/10 PING OK - Packet loss = 0%, RTA = 2.00 ms
dccois3 LDAP Service OK 2012-07-24 04:56:22 0d 14h 15m 34s 1/10 LDAP OK - 0.022 seconds response time
PING OK 2012-07-24 04:57:26 3d 12h 0m 30s 1/10 PING OK - Packet loss = 0%, RTA = 1.29 ms
dccois4 LDAP Service OK 2012-07-24 04:57:28 0d 13h 44m 28s 1/10 LDAP OK - 0.027 seconds response time
PING OK 2012-07-24 04:57:36 7d 19h 30m 20s 1/10 PING OK - Packet loss = 0%, RTA = 2.87 ms
dccois5 PING OK 2012-07-24 04:56:41 11d 4h 9m 15s 1/10 PING OK - Packet loss = 0%, RTA = 0.26 ms
dccois6 Domain Name Service OK 2012-07-24 04:56:31 2d 4h 11m 25s 1/10 DNS OK: 0.016 seconds response time. dccois5.INDIANOIL.IN returns 10.52.2.43
LDAP Service OK 2012-07-24 04:57:33 1d 8h 50m 23s 1/10 LDAP OK - 0.009 seconds response time
PING OK 2012-07-24 04:56:34 2d 4h 13m 22s 1/10 PING OK - Packet loss = 0%, RTA = 0.17 ms
dref PING OK 2012-07-24 04:56:14 0d 11h 57m 42s 1/10 PING OK - Packet loss = 0%, RTA = 48.10 ms
erpprd PING OK 2012-07-24 04:57:37 0d 14h 4m 19s 1/10 PING OK - Packet loss = 0%, RTA = 2.49 ms
gref PING OK 2012-07-24 04:57:14 0d 11h 12m 42s 1/10 PING OK - Packet loss = 0%, RTA = 41.00 ms

Zenoss


Zenoss Enterprise
Device/IP Search
Search by device name or IP address
admin Preferences Logout Help
Zenoss server time: 12:40:01
Click to toggle menu visibility
Configure layout... Add portlet... Stop Refresh
Main Views
Dashboard
Event Console
Device List
Network Map
Classes
Events
Devices
Services
Processes
Products
Browse By
Systems
Groups
Locations
Networks
Reports
Management
Add Device
Mibs
Collectors
Settings
Event Manager
Navigation Menu
Navigate Zenoss views and select tasks
Locations
Root Organizers
Manage and configure the dashboard
Object
Events
/Systems/Testing
/Systems/Trading
/Systems/BlahBlahBlah
/Systems/Network
/Systems/Buildbot
/Systems/CRM
/Systems/Internet
Root Organizers
Configure this portlet
Object
Events
/Groups/Admin 1 Group
/Groups/Support
/Groups/build
/Groups/Network
/Groups/Customers
Map data ©2008 LeadDog Consulting, Tele Atlas, Europa Technologies - Terms of Use

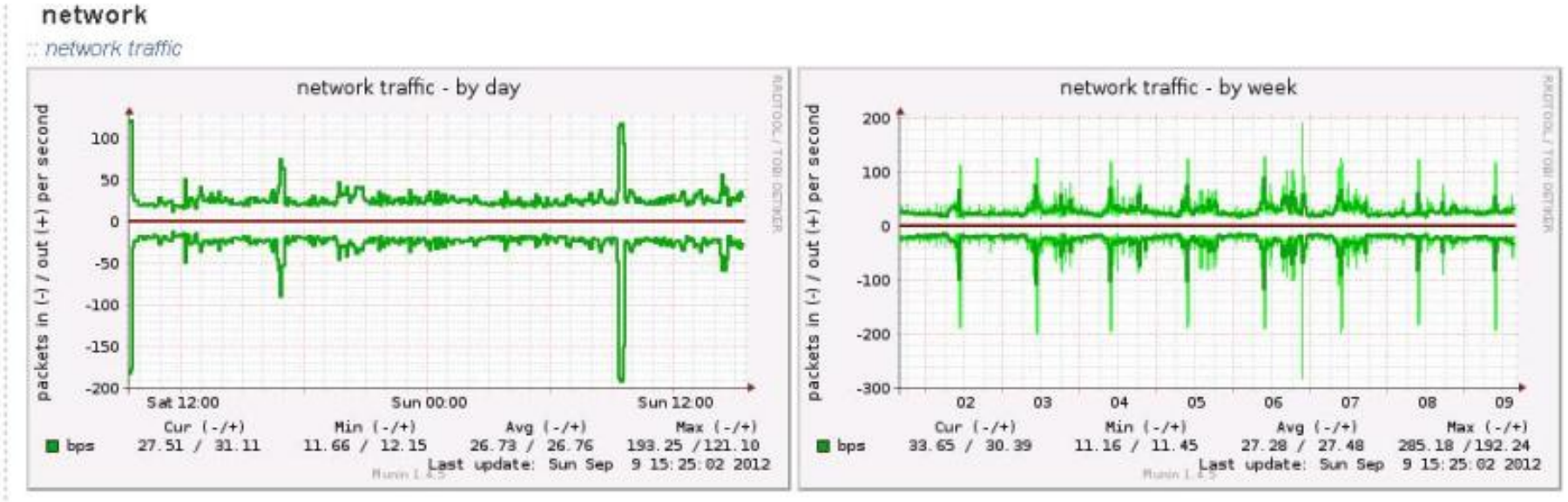
Icinga


Icinga – Downtimes
Monitoring
Admin
Help
Servertime: 06:27:00
Press CTRL+ALT+F ...
Root, Enoch
26 / 0 UP
0 / 4 / 0 DOWN
0 / 0 / 0 UNREACHABLE
0 PENDING
4 / 30 IN TOTAL
32 / 3 OK
0 / 0 / 0 WARNING
2 / 4 / 4 CRITICAL
0 / 0 / 0 UNKNOWN
0 PENDING
10 / 45 IN TOTAL
1 OK
0 DOWN
Welcome
ServiceStatus
Overview
Downtimes
Reload
Categories
Settings
Refresh
Settings
Filter
Commands
My Category (2)
Data (13)
Unhandled se...
Unhandled ho...
Open problems
ServiceStatus
ServiceHistory
HostStatus
HostHistory
Hostgroups
Servicegroups
Downtimes
Notifications
Status Map
Instances
Tactical Overview (3)
Reporting (2)
Business Process (2)
Misc (7)
Host
Host status
Service
Service status
Scheduled start
Scheduled end
Author
Comment
web_de-pop POP3 CRITICAL 2012-05-16 17:23:19 2012-05-16 19:25:19 guest 123
google-www HTTP CRITICAL 2012-03-07 16:42:57 2037-03-07 18:42:57 guest it's doooooown
google-www HTTP CRITICAL 2012-04-03 14:57:36 2012-04-03 16:58:36 guest ,kboiksoidhhosa sd
google-www HTTP CRITICAL 2012-05-16 17:23:19 2012-05-16 19:25:19 guest 123
web_de-pop UP 2012-04-02 11:55:49 2012-04-02 13:55:49 root comment
web_de-pop UP 2012-05-16 17:21:26 2012-05-16 17:25:26 guest Test
web_de-pop UP 2014-05-05 11:19:21 2014-05-05 13:19:21 guest x
c2-printserver-1 UP 2012-04-02 11:57:02 2012-04-02 13:57:02 root comment Not running
gmx-pop POP3 CRITICAL 2011-12-21 01:46:39 2011-12-21 03:46:39 guest Die Kist eist Down, muss ...
gmx-pop POP3 CRITICAL 2012-02-13 18:46:01 2012-02-13 20:46:01 guest .
gmx-pop POP3 CRITICAL 2012-02-26 21:40:39 2012-02-26 23:40:39 guest fhgjghj
gmx-pop POP3 CRITICAL 2012-03-22 13:53:38 2012-03-22 15:53:38 guest test
gmx-pop POP3 CRITICAL 2012-04-23 09:42:22 2012-05-23 00:00:00 guest doin stuff and such
gmx-pop POP3 CRITICAL 2012-05-16 17:23:19 2012-05-16 19:25:19 guest 123
web_de-www HTTP CRITICAL 2012-01-31 12:10:00 2012-01-31 13:00:00 Admin Test
Page 1 of 6
Displaying topics 1 - 25 of 133

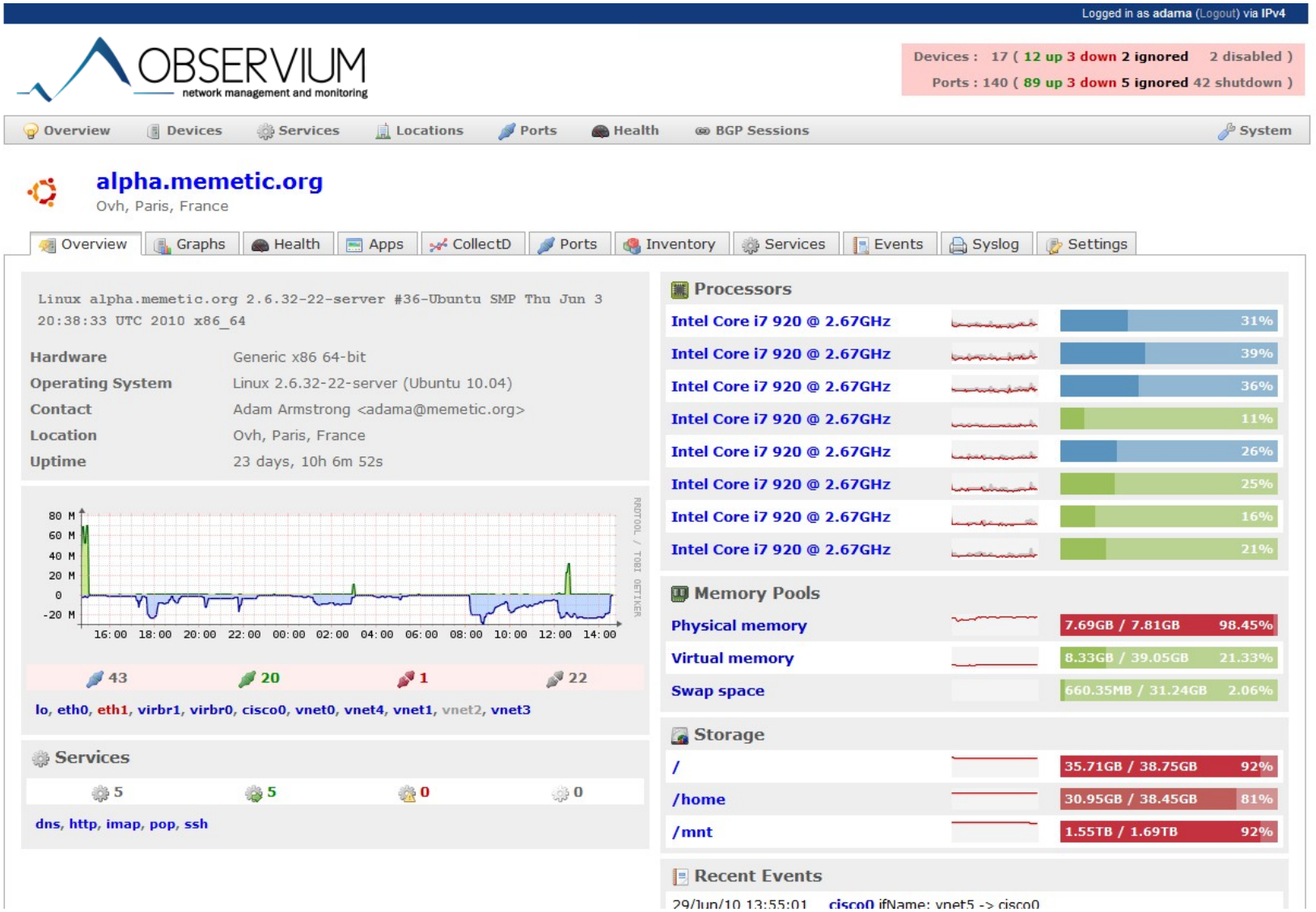
Separate løsninger

- Trafikk trender
- SLA

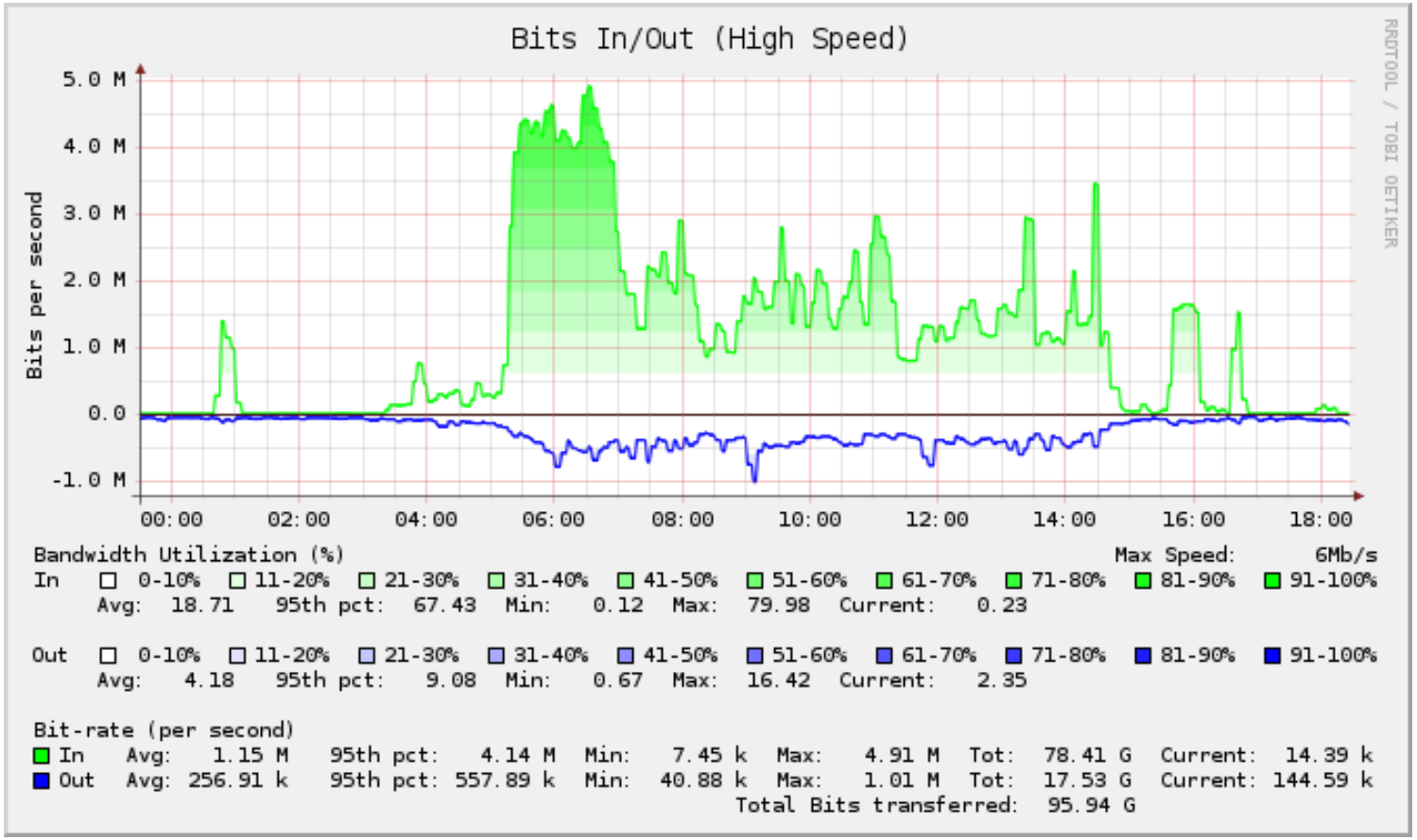
Munin


network
network traffic
network traffic - by day
packets in (-) / out (+) per second
100
50
0
-50
-100
-150
-200
Sat 12:00
Sun 00:00
Sun 12:00
Cur (-/+)
Min (-/+)
Avg (-/+)
Max (-/+)
bps
27.51 / 31.11
11.66 / 12.15
26.73 / 26.76
193.25 /121.10
Last update: Sun Sep 9 15:25:02 2012
network traffic - by week
packets in (-) / out (+) per second
200
100
0
-100
-200
-300
02
03
04
05
06
07
08
09
Cur (-/+)
Min (-/+)
Avg (-/+)
Max (-/+)
bps
33.65 / 30.39
11.16 / 11.45
27.28 / 27.48
285.18 /192.24
Last update: Sun Sep 9 15:25:02 2012

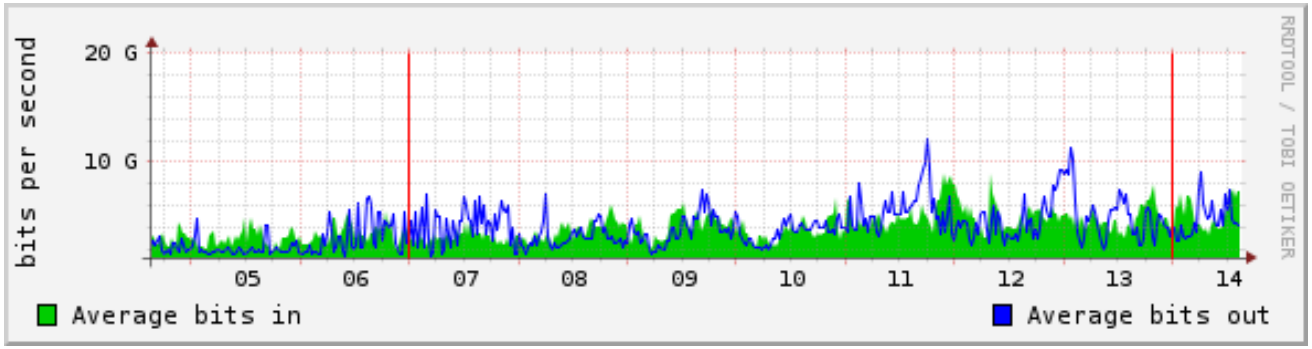
Observium


Logged in as adama (Logout) via IPv4
OBSERVIUM
network management and monitoring
Devices : 17 (12 up 3 down 2 ignored 2 disabled)
Ports : 140 (89 up 3 down 5 ignored 42 shutdown)
Overview
Devices
Services
Locations
Ports
Health
BGP Sessions
System
alpha.memetic.org
Ovh, Paris, France
Overview
Graphs
Health
Apps
CollectD
Ports
Inventory
Services
Events
Syslog
Settings
Linux alpha.memetic.org 2.6.32-22-server #36-Ubuntu SMP Thu Jun 3 20:38:33 UTC 2010 x86_64
Hardware
Generic x86 64-bit
Operating System
Linux 2.6.32-22-server (Ubuntu 10.04)
Contact
Adam Armstrong <adama@memetic.org>
Location
Ovh, Paris, France
Uptime
23 days, 10h 6m 52s
80 M
60 M
40 M
20 M
0
-20 M
16:00 18:00 20:00 22:00 00:00 02:00 04:00 06:00 08:00 10:00 12:00 14:00
43 20 1 22
lo, eth0, eth1, virbr1, virbr0, cisco0, vnet0, vnet4, vnet1, vnet2, vnet3
Services
5 5 0 0
dns, http, imap, pop, ssh
Processors
Intel Core i7 920 @ 2.67GHz 31%
Intel Core i7 920 @ 2.67GHz 39%
Intel Core i7 920 @ 2.67GHz 36%
Intel Core i7 920 @ 2.67GHz 11%
Intel Core i7 920 @ 2.67GHz 26%
Intel Core i7 920 @ 2.67GHz 25%
Intel Core i7 920 @ 2.67GHz 16%
Intel Core i7 920 @ 2.67GHz 21%
Memory Pools
Physical memory 7.69GB / 7.81GB 98.45%
Virtual memory 8.33GB / 39.05GB 21.33%
Swap space 660.35MB / 31.24GB 2.06%
Storage
/ 35.71GB / 38.75GB 92%
/home 30.95GB / 38.45GB 81%
/mnt 1.55TB / 1.69TB 92%
Recent Events
29/Jun/10 13:55:01 cisco0 ifName: vnet5 -> cisco0

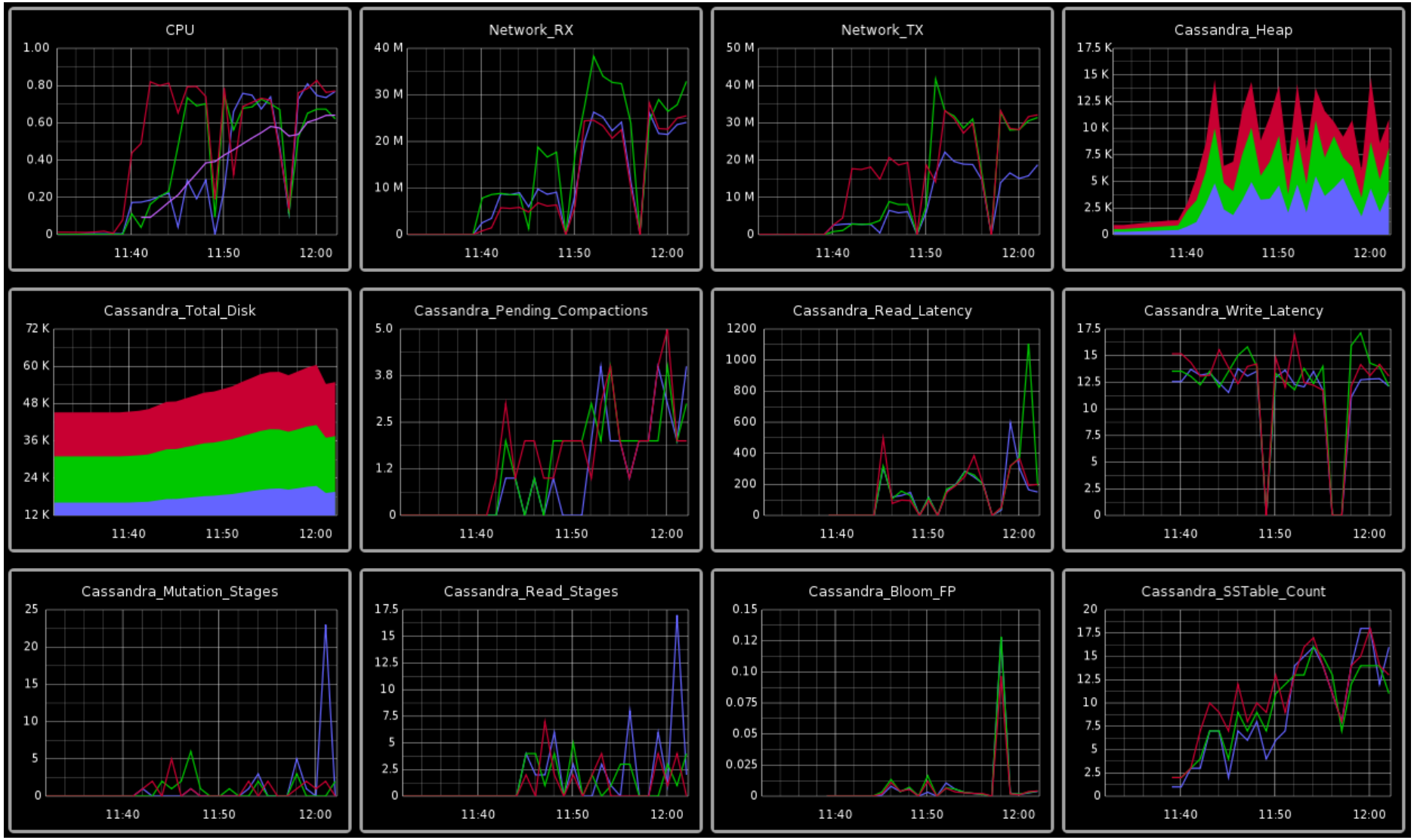
RRD-Tool


Bits In/Out (High Speed)
Bits per second
5.0 M
4.0 M
3.0 M
2.0 M
1.0 M
0.0
-1.0 M
00:00
02:00
04:00
06:00
08:00
10:00
12:00
14:00
16:00
18:00
RRDTOOL / TOBI OETIKER
Bandwidth Utilization (%)
Max Speed: 6Mb/s
In 0-10% 11-20% 21-30% 31-40% 41-50% 51-60% 61-70% 71-80% 81-90% 91-100%
Avg: 18.71 95th pct: 67.43 Min: 0.12 Max: 79.98 Current: 0.23
Out 0-10% 11-20% 21-30% 31-40% 41-50% 51-60% 61-70% 71-80% 81-90% 91-100%
Avg: 4.18 95th pct: 9.08 Min: 0.67 Max: 16.42 Current: 2.35
Bit-rate (per second)
In Avg: 1.15 M 95th pct: 4.14 M Min: 7.45 k Max: 4.91 M Tot: 78.41 G Current: 14.39 k
Out Avg: 256.91 k 95th pct: 557.89 k Min: 40.88 k Max: 1.01 M Tot: 17.53 G Current: 144.59 k
Total Bits transferred: 95.94 G

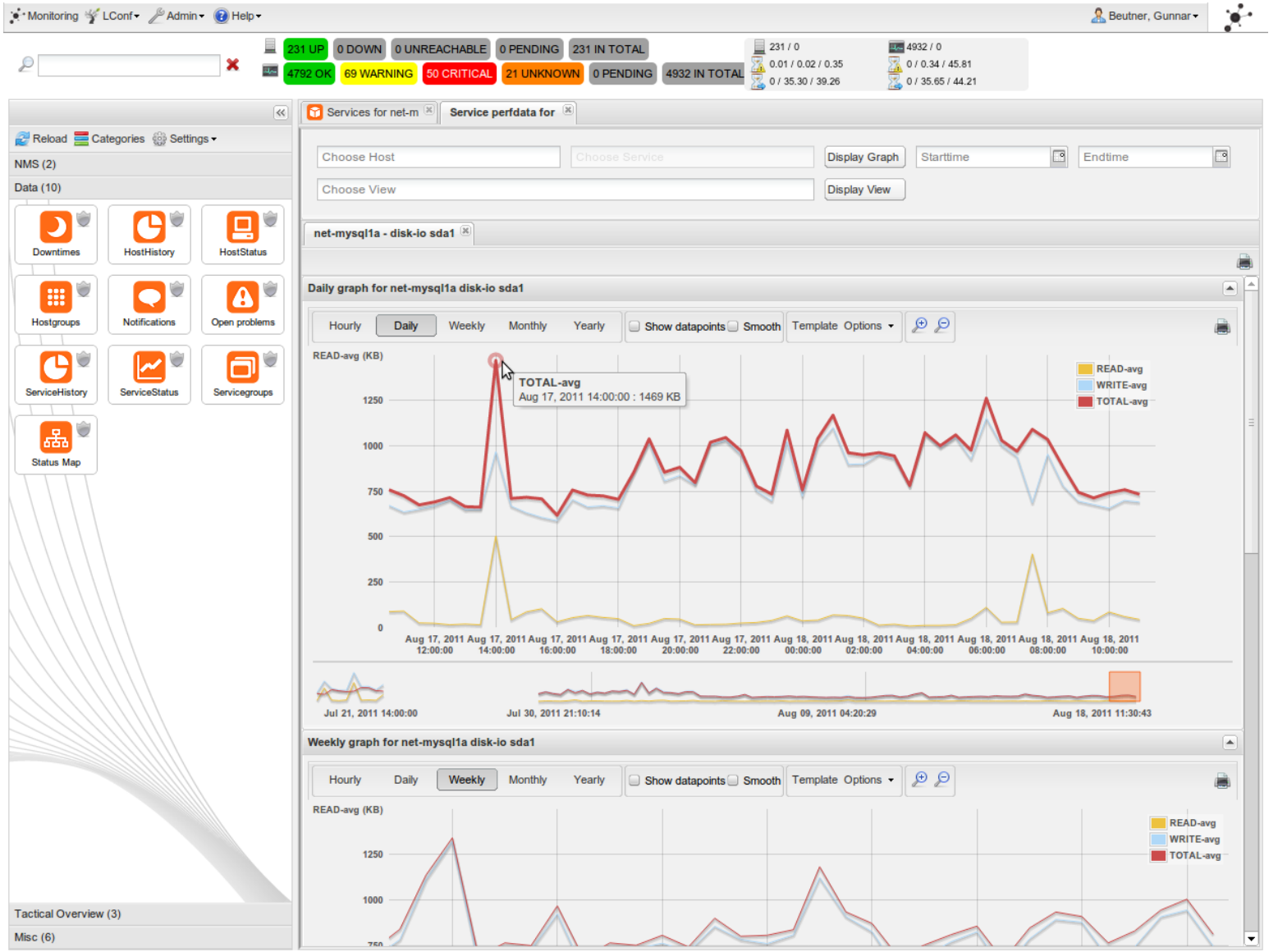
MRTG


bits per second
20 G
10 G
05
06
07
08
09
10
11
12
13
14
Average bits in
Average bits out
RRDTOOL / TOBI OETIKER

Graphite


CPU
1.00
0.80
0.60
0.40
0.20
0
11:40
11:50
12:00
Network_RX
40 M
30 M
20 M
10 M
0
11:40
11:50
12:00
Network_TX
50 M
40 M
30 M
20 M
10 M
0
11:40
11:50
12:00
Cassandra_Heap
17.5 K
15 K
12.5 K
10 K
7.5 K
5 K
2.5 K
0
11:40
11:50
12:00
Cassandra_Total_Disk
72 K
60 K
48 K
36 K
24 K
12 K
11:40
11:50
12:00
Cassandra_Pending_Compactions
5.0
3.8
2.5
1.2
0
11:40
11:50
12:00
Cassandra_Read_Latency
1200
1000
800
600
400
200
0
11:40
11:50
12:00
Cassandra_Write_Latency
17.5
15
12.5
10
7.5
5
2.5
0
11:40
11:50
12:00
Cassandra_Mutation_Stages
25
20
15
10
5
0
11:40
11:50
12:00
Cassandra_Read_Stages
17.5
15
12.5
10
7.5
5
2.5
0
11:40
11:50
12:00
Cassandra_Bloom_FP
0.15
0.12
0.10
0.075
0.05
0.025
0
11:40
11:50
12:00
Cassandra_SSTable_Count
20
17.5
15
12.5
10
7.5
5
2.5
0
11:40
11:50
12:00

inGraph


Monitoring
LConf
Admin
Help
Beutner, Gunnar
231 UP
0 DOWN
0 UNREACHABLE
0 PENDING
231 IN TOTAL
4792 OK
69 WARNING
50 CRITICAL
21 UNKNOWN
0 PENDING
4932 IN TOTAL
Reload
Categories
Settings
NMS (2)
Data (10)
Downtimes
HostHistory
HostStatus
Hostgroups
Notifications
Open problems
ServiceHistory
ServiceStatus
Servicegroups
Status Map
Tactical Overview (3)
Misc (6)
Services for net-m
Service perfdata for
Choose Host
Choose Service
Display Graph
Starttime
Endtime
Choose View
Display View
net-mysql1a - disk-io sda1
Daily graph for net-mysql1a disk-io sda1
Hourly
Daily
Weekly
Monthly
Yearly
Show datapoints
Smooth
Template Options
READ-avg (KB)
TOTAL-avg
Aug 17, 2011 14:00:00 : 1469 KB
READ-avg
WRITE-avg
TOTAL-avg
Weekly graph for net-mysql1a disk-io sda1

Icinga

MonKey Presentasjon 29.01.2013

Monitoring is Key

Valg av kjerneprogramvare (Icinga)

- Fork av nagios
  - Nagios-kompatibel
- Bra dokumentasjon
- Icinga reporting
- Icinga web
- Icinga mobile
- API mot Icinga Core
- Icinga 2

Objekter

- Host
- Hostgroup
- Service
  - command + host/hostgroup
- Servicegroup
- Command
- Host- og service dependency
- Contact
- Contactgroup
- Timeperiod
- Host- og service escalation

Templates og arv

```
define host {
        check_command           check-host-alive
        notification_options    d,u,r
        max_check_attempts      5
        name                    generichosttemplate
        register                0
 }

 define host {
        host_name               bighost1
        address                 192.168.1.3
        use                     generichosthosttemplate
}
```

Templates og arv

- Selvdefinerte variabler f.eks til ssh port.
- Kan arve i flere nivåer
- Overstyres i den konfigurasjonen som er mest lokal.

Konfigurasjonsfiler

- Bygget på tekstfiler
- Leter rekursivt i /etc/icinga/objects/*.cfg
- Sjekk med icinga -v icinga.cfg
- service icinga reload
- Konfigurasjonen lagres i MySQL/PostgreSQL db.

Legge til nye hosts

- cp template_host ns03.cnf
- nconf
- Lage en enkel "velg hostgroups for ny host" webgui?

Webgrensesnitt

Icinga classic

Icinga-Web

Generell overvåkning

- DHCP
- DNS
- LDAP
- SMB
- MySQL
- HTTP

HiG3	Check DHCP	OK	2013-01-28 14:26:31	2d 11h 13m 45s	1/4	OK: Received 1 DHCPOFFER(s), 1 of 1 requested servers responded, max lease time = 0 sec.
	Check DNS	OK	2013-01-28 14:26:53	1d 0h 23m 23s	1/4	DNS OK: 0.012 seconds response time. hig3.monkey.local returns 10.60.0.22
	Check LDAP	OK	2013-01-28 14:26:43	3d 1h 9m 15s	1/4	LDAP OK - 0.004 seconds response time
	Check SMB	OK	2013-01-28 14:29:05	2d 23h 46m 11s	1/4	Disk ok - 19.71G (49%) free on \10.60.0.22testshare

HiG1	Check HTTP	OK	2013-01-28 14:23:58	3d 0h 2m 40s	1/4	HTTP OK: HTTP/1.1 200 OK - 453 bytes in 0.001 second response time
	Check MySQL DB	OK	2013-01-28 14:24:43	3d 0h 1m 55s	1/4	Uptime: 523832 Threads: 3 Questions: 434659 Slow queries: 0 Opens: 1066 Flush tables: 2 Open tables: 64 Queries per second avg: 0.829

Overvåkning av Linux-servere

- SSH
  - Må installere plugins lokalt på hoster som overvåkes
  - ssh-key login for en overvåkningsbruker
- SNMP
- NRPE (Nagios Remote Plugin Executor)
  - Mulighet for SSL

Host	Service	Status	Last Check	Duration	Attempt	Status Information
HiG1	Check HTTP	OK	2013-01-28 14:38:58	3d 0h 15m 19s	1/4	HTTP OK: HTTP/1.1 200 OK - 453 bytes in 0.001 second response time
	Check MySQL DB	OK	2013-01-28 14:34:43	3d 0h 14m 34s	1/4	Uptime: 524432 Threads: 3 Questions: 434993 Slow queries: 0 Opens: 1066 Flush tables: 2 Open tables: 64 Queries per second avg: 0.829
	Current Load	OK	2013-01-28 14:37:08	2d 23h 32m 52s	1/4	OK - load average: 0.01, 0.02, 0.00
	Current Users	OK	2013-01-28 14:36:55	3d 0h 13m 4s	1/4	USERS OK - 1 users currently logged in
	Disk Space	OK	2013-01-28 14:36:58	3d 0h 12m 19s	1/4	DISK OK
	SSH	OK	2013-01-28 14:37:43	3d 0h 11m 34s	1/4	SSH OK - OpenSSH_5.5p1 Debian-6+squeeze2 (protocol 2.0)
	Total Processes	OK	2013-01-28 14:39:10	3d 0h 10m 49s	1/4	PROCS OK: 71 processes
	ido2db Process	OK	2013-01-28 14:34:50	3d 0h 15m 4s	1/3	PROCS OK: 2 processes with command name 'ido2db'
HiG2	Check HTTP	OK	2013-01-28 14:39:09	3d 1h 20m 46s	1/4	HTTP OK: HTTP/1.1 200 OK - 453 bytes in 0.001 second response time
	Check MySQL DB	OK	2013-01-28 14:34:16	3d 1h 20m 1s	1/4	Uptime: 534781 Threads: 1 Questions: 20280 Slow queries: 0 Opens: 1160 Flush tables: 1 Open tables: 64 Queries per second avg: 0.37
	Current Load	OK	2013-01-28 14:37:30	2d 22h 22m 37s	1/4	OK - load average: 0.00, 0.00, 0.00
	Current Users	OK	2013-01-28 14:37:45	3d 1h 18m 31s	1/4	USERS OK - 0 users currently logged in
	Disk Space	OK	2013-01-28 14:35:50	2d 21h 58m 27s	1/4	DISK OK
	SSH	OK	2013-01-28 14:37:16	3d 1h 17m 1s	1/4	SSH OK - OpenSSH_5.5p1 Debian-6+squeeze2 (protocol 2.0)
	Total Processes	OK	2013-01-28 14:34:40	2d 4h 44m 37s	1/4	PROCS OK: 58 processes

Overvåking av Windows-servere

- NSClient++
  - Wrapper for check_nt, NRPE og NSCA
- NC_Net
- checkwmiplus
  - krever WMI-les-rettigheter på hosts

Host	Service	Status	Last Check	Duration	Attempt	Status Information
HiG3	CPU Load	OK	2013-01-28 14:28:46	3d 1h 14m 30s	1/4	CPU Load 0% (5 min average)
	Disk space	OK	2013-01-28 14:31:46	3d 1h 11m 30s	1/4	c: - total: 39.90 Gb - used: 20.18 Gb (51%) - free 19.72 Gb (49%)
	Memory usage	OK	2013-01-28 14:32:31	3d 1h 10m 45s	1/4	OK memory within bounds.
	Print Spooler	OK	2013-01-28 14:32:44	2d 23h 20m 32s	1/4	spoolsv.exe: Running
	RDP-Sessions: Active	OK	2013-01-28 14:28:41	2d 23h 1m 34s	1/4	0
	RDP-Sessions: Inactive	OK	2013-01-28 14:29:18	2d 23h 10m 5s	1/4	2

Server og infrastruktur

MonKey Presentasjon 05.03.2012

Monitoring is Key

Status

- Server
  - Prosesser (Print Spooler, DNS, DHCP og Apache)
  - Exchange
  - Tjenester (LDAP / DHCP / DNS / HTTP)
  - SQL (MySQL, MSSQL og Oracle)
  - Gruppere tjenester sammen (Apache + MySQL)
  - Counters (Terminal services, Exchange etc.)
- Infrastruktur
  - Cisco-brannmur / Switch (helse, failover, load)
  - Dell, HP Switcher (helse)
  - APC / HP UPS (spenning, load, charge, temp)
  - XEN / VMWare (helse status, load)
  - Meru Wireless (status, load)

Konfigurasjon

- Sjekker basert på hostgroup
  - Generiske og spesifikke
  - Eks: windows_servers, mysql_servers
- Parent / child
- Check_multi

Vedlegg D

Dokumenter

I dette vedlegget finnes:
Kartlegging av behov, som ble laget av MonKey og utredet sammen med IKT-avdelingen.
Opplæringsdokument som ble gjennomgått sammen med Oppdragsgiver og Teknisk kontakt ved slutten av prosjektperioden.

Kartlegging av behov

Tirsdag 19. Mars 2013

Sted: Møterom IKT
Tilstede:
Fra MonKey: Bjørn-Erik Strand og Nils Slåen
Fra IKT-avdelingen: Tore Vegard, Are Stenbrenden og Kristoffer Rødsdalen Hagen

På starten av møtet ble det diskutert om muligheten for å prioritere feil, som i en skala fra 1 og opp til 10. Denne skulle da kunne regnes ut ifra antall children til hosten, antall service sjekker, prioritet på servicesjekken, og prioritering som er manuelt satt på hver host.

1 Overvåkning

Hvordan bruker dere overvåkningen til daglidagse oppgaver: Det som er ønskelig er å bruke overvåkningen aktivt, det vil si at når det dukker opp feil på overvåkningen vil personale på servicedesk ta tak i problemet med en gang, enten lage en sak på dette som blir videre eskalert, eller løse feilen der og da.

Kjenner dere et behov for å bruke overvåkningen mer aktivt fra dag til dag, for å avdekke behov for oppgradering, vedlikehold, utskifting av utstyr: I dagens løsning er det vanskelig å synliggjøre om tjenesten og /eller hosten skal være nede eller ikke her skulle det vært muligheter for å schedule unscheduled downtime, slik at det fungerer i praksis.

IKT-avdelingen har også et ønske om mer realtime overvåkning, dagens løsning oppdaterer seg for sent, de har også et ønske om at ressursforbruk blir bedre logget, og at man kan gå tilbake for å se historikk for dette, slik at underliggende feil kan avdekkes.

Hvor ofte blir det satt opp utstyr der det ville være natulig å legge til enheten til overvåkning: I perioder blir det ikke satt opp mye nytt utstyr, men det blir noe utbytting av gamle servere og utstyr. Det er mest på prosjekter at nye enheter settes i produksjon, her er IKT-avdelingen bevisste på å legge til disse enhetene i overvåkningen.

Settes det opp utstyr som burde overvåkes bedre (Flere tjenester som overvåkes etc): Alle mener her at det ikke er nok tjenester som overvåkes i dag, en av hovedgrunnen til dette er at siden de ikke er fornøyde med den eksisterende overvåkningsløsningen, er det kun de "viktigste" sjekkene som blir lagt inn. Også blir dagens overvåkningsskjerm ofte berørt av for mye informasjon, og det er vannskelig å sile ut hva som er feil. Det blir altså vekk kastet tid å implementere flere sjekker som tilsynelatende ikke fungerer slik de skal.

Hvilke rutiner følges i dag for vedlikehold av både eksisterende og nye enheter i overvåkningen: Før ble det brukt ITIL change rutiner for å legge til nytt utstyr, disse rutinene krevde godkjenning av change manager for å få klarsignal til produksjon, disse rutinene er ikke lengre et krav og enkelte service sjekker kan da bli glemt før det implementeres. Derfor får IKT-avdelingen ofte manglende overvåkning av spesifikke applikasjoner. Kan også nevnes at alle er flinke på å legge til hosts i overvåkningen. Ønsker rutiner for tresholds slik at man kan finjustere disse.

2 Gjentagene feil

Hva skulle dere ønske at kunne oppdage raskt: I dag kjøres alle sjekkene på lik linje, her må man tenke på at skolene er 80% av brukerne, så i det tilfellet at en skole går ned vil IKT-avdelingen ha melding om dette umiddelbart, dette burde også eskaleres, og sjekkinterval høyre enn vanlige hosts. Det er også et forslag at man skal kunne ha ekstra overvåkning i visse perioder, som eksamen.

Er det stort sett feil dere har sett før eller nye feil: IKT-avdelingen ser ofte gjentagende feil som CPU, Minne, Ping timeouts osv, det vise rødt på overvåkningen og er irriterende at det viser som feil, når det kanskje ikke er det.

Så det er stort sett feil som går igjen.

Tror dere det forekommer mange feil som dagens overvåkningsløsning ikke oppdager: Det IKT-avdelingen er helt sikker på at det forekommer mange feil som i dag ikke fanges opp, dette er ofte heller oppdaget av at brukerne ringer inn. Her er det ønskelig at flere applikasjoner overvåkes, og at man kan se på trender over utstyr og oppetid.

3 Hva ønsker dere å få vite

Hvordan burde et evt varslingsgrensesnitt se ut? Ingen informasjon når alt er OK, informasjon om load og trafikk etc? Kan dette evt. ligge på en egen "webside" (icinga-web-ish): Ønsker å se totalt antall nede, og hvilke som er nede, vist mest mulig minimalistisk, slik at det er lett å holde oversikten, og man får "beskjed" dersom det skjer noe nytt.

Hvordan vil dere bli varslet utenom visning på skjerm: IKT-avdelingen kan ikke bli pålagt å ha slik varsling, her blir det eventuellt den som har vakt, eller kan det være ønskelig for de som har ansvaret for spesifikke systemer å få en melding om noe skulle være galt.

Det et ønske å ha oversikt over båndbreddebruk på spesifikke porter på nettverksutstyr. Disse kunne gjerne vært integrert som trafikkgrafer i statusvisningen.

Oppgaver for bruk av Icinga

Oppgaver utenom Icinga

Arbeidsoppgaver i Windows:

- Installere NSClient++
- Konfigurere NSClient++
- Finne tjenestenavn, legge til custom plugins

Arbeidsoppgaver i Linux

- Installere NRPE med plugins for Linux
- Konfigurere NRPE for Linux

Arbeidsoppgaver på annet nettverksutstyr

- Konfigurere SNMP
- Finne OID til MiB, via SNMPWalk

Oppgaver i Icinga

“Dagligdagse oppgaver”

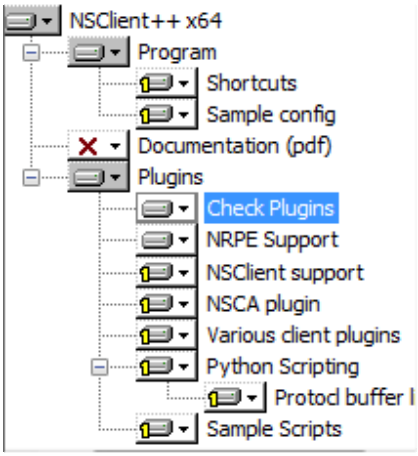
- Legge til ny Hosts
    - Hostname (FQDN for host)
    - Address (IP adressen til hosten)
    - Alias (Samme som hostname)
    - Parents (Switch denne er koblet på feks)
    - Hostgroups (Hvilke sjekker skal kjøres, hva gjør denne serveren)
- Legge til ny Services (Etter host er lagt til hvor service ikke eksisterer)
    - Nye windows tjenester (Hva heter den nye tjenesten?)
    - Andre parametre i eksisterende servicer (SSH port, andre thresholds)
    - Bruke eksisterende check command (SNMP kan brukes av mange forskjellige)
- Legge til ny Hostgroup (Etter host er lagt til hvor hostgroup for service ikke eksisterer)
    - Hostgroup_navn (Navnet som skal referes til)
    - Alias
- Reload av konfig (Håntere feilmeldinger)
- Navigere i Icinga GUI
    - Reschedule sjekker (Endring gjøres og man ønsker å kjøre sjekken på nytt)
    - Acknowledge alarmer (En alarm vises men tas hånd om)
    - Se konfigurasjon
    - Se relasjon mellom performance data og grafer (Hva sjekkene gir tilbake)

“Andre oppgaver i Icinga”

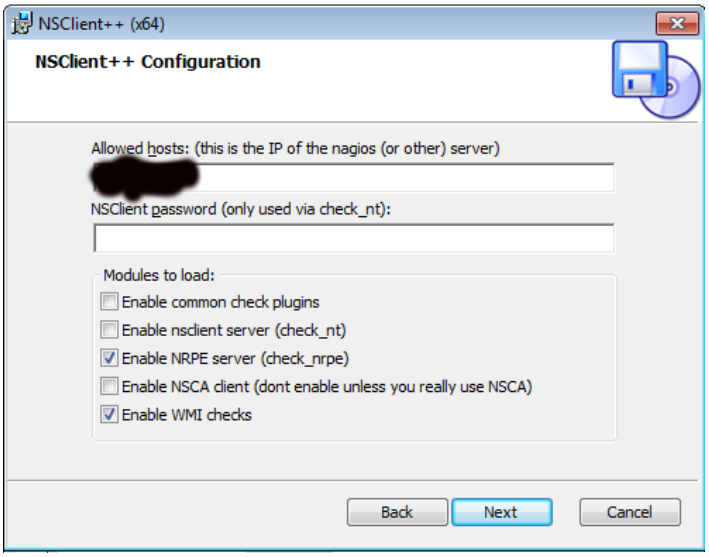
- Gjøre seg kjent med mappestruktur
    - Hvor ligger config
    - Hva heter de forskjellige konfig filene
- Legge til ny plugin
    - Riktig thresholds
    - Sende riktig argument
    - Bruke macro
- Legge til ny kommando
    - Ta imot riktig argument
    - Bruke macro
- Restart av Icinga server
- Endring av passord
    - Icinga-classic web (htpasswd -c /etc/icinga/htpasswd.users icingaadmin)
    - Icinga-web
- LDAP auth
    - Icinga-classic web
    - icinga-web
- Vedlikehold av kontakter
    - Holde kontaktene oppdatert
- Vedlikehold og oppretting av generics
    - Lage egne tidsrammer for varsling
    - Ha hosts og eller services som sjekkes oftere
    - Legge til kontaktinfo
- Vedlikehold og oppretting av ext info for hosts.
    - Bilder i icinga classic GUI
- Statusvindu
    - Legge til nye sensorer

NSClient++ NRPE Guide

Last ned NSClient fra: http://www.nsclient.org/nscp/downloads



Start MSI filen, velg så bort alt bortsett fra Check Plugins og NRPE Support.



Skriv inn IP adressen til Icinga serveren, fjern alt bortsett fra NRPE server og WMI checks.
Kopier konfig fil fra sentral server. Verifiser at allow_nasty_meta_chars er på.

Linux NRPE Guide

Under følger installasjonsinstruks for NRPE på linux-distroer som benytter apt (ie. debian-varianter og avarter).

apt-get install nagios-plugins-basic libnagios-plugin-perl nagios-nrpe-server
cd /usr/lib/nagios/plugins && mkdir libexec
cd libexec && wget https://raw.github.com/jasonhancock/nagios-memory/master/plugins/check_mem --no-check-certificate
chmod +x check_mem

vim /etc/nagios/nrpe.cfg

log_facility=daemon
pid_file=/var/run/nagios/nrpe.pid
server_port=5666
nrpe_user=nagios
nrpe_group=nagios
allowed_hosts=127.0.0.1,%IP ADRESSE TIL ICINGA%
dont_blame_nrpe=1
debug=0
command_timeout=60
connection_timeout=300
include=/etc/nagios/nrpe_local.cfg
include_dir=/etc/nagios/nrpe.d/

vim /etc/nagios/nrpe_local.cfg

command[check_all_mounts]=/usr/lib/nagios/plugins/check_disk -w $ARG1$ -c $ARG2$ --mountpoint --all
command[check_dist_load]=/usr/lib/nagios/plugins/check_load -r -w $ARG1$ -c $ARG2$
command[check_mem]=/usr/lib/nagios/plugins/libexec/check_mem.pl -w $ARG1$ -c $ARG2$
command[check_process]=/usr/lib/nagios/plugins/check_procs -C $ARG1$ -c $ARG2$
command[check_apt]=/usr/lib/nagios/plugins/check_apt --include=$ARG1$ --exclude=$ARG2$ --critical=$ARG3$
command[check_users]=/usr/lib/nagios/plugins/check_users -w $ARG1$ -c $ARG2$

Vedlegg E

Statusrapporter

I dette vedlegget finnes statusrapporter skrevet for hver iterasjon.

MonKey: Statusrapport

Onsdag 13. Februar 2013

Første iterasjon: Kjerneprogramvare

Vi endte etter en del research opp med å basere oss på Icinga. Vi brukte så tid på å lese dokumentasjonen og teste icinga virtualisert på egne maskiner. Da vi fikk opp labmiljøet på fylkeshuset ble testingen overført dit.

Vi brukte en god del tid på å finne ut hvordan en best mulig kan gjøre endringer i konfigurasjon. Et par forskjellige webgrensesnittt ble testet ut. Men vi synes ikke at noen av de dekker våre behov og har konkludert med at det beste er å bruke tekstkonfigurasjonen direkte. Det største problemet med webgrenesnitt-løsningene er at de er at de ikke tillater oppdeling av konfigurasjonen og en mister dermed lett oversikten. Det viste seg også vanskelig å kombinere konfigurasjon manuelt fra CLI med webgrensesnittene. Eksisterende filer ble dessuten overskrevet ved hver import, noe som absolutt ikke er ønskelig i et produksjonsmiljø.

Så langt har vi fått satt opp en labserver på fylkeshuset der vi har lagt inn Icinga med overvåking av 1 Linux- og 1 Windows-server

På iterasjonsmøte med Lasse og Svein-Inge fikk vi klarsignal til å gå videre med Icinga som kjernemodul. Så langt har ting gått som planlagt og vi er godt i rute.

MonKey: Statusrapport

Onsdag 20. Februar 2013

Andre iterasjon: Server

Vi begynte iterasjonen med grunnsjekker for Windows og Linux-servere. Dette er i hovedsak hardware som RAM, disk og CPU. En del tid har gått med på å finne et fornuftig oppsett for konfigurasjon av maskiner og tjenester. For å kunne se samlet status av en tjeneste (f.eks for et cluster) har vi testet sjekker som check_multi og check_cluster.

Vi har lest om og tester ulike protokoller for kjøring av sjekker på eksterne maskiner. Valget endte til slutt på NRPE for både Linux og Windows. Andre alternativer vi vurderte var SSH, SNMP og WMI.

For applikasjoner har vi hatt fokus på å teste spesielle applikasjoner som krever litt mer enn at bare en service kjører. Her har vi blant annet sett på hvordan vi ser helsestatus til SQL-servere og ytelse på exchange server. For andre applikasjoner har vi funnet en generisk måte å definere de som et sett tjenester, og hvordan disse kan overvåkes.

Videre har vi sett på avhengigheter og parent/child-forhold som også vil gjelde for nettverksutstyr. Dette er viktig å ha på plass når vi skal begynne med varsling slik at en ikke får mange varlser for feil som blir slått som følge av mer underliggende feil.

På iterasjonsmøte fikk vi tilbakemeldinger av Svein-Inge og Lasse på hvordan de ville prioritere de forskjellige applikasjonene vi skal monitorere. Dette gjorde det lettere å sette opp en plan for de siste dagene av iterasjonen.

Status så langt for selve overvåkningen er at vi har satt opp MSSQL og Exchange til overvåkning i lab. Vi ble ikke helt ferdig med Exchange og tar med denne i neste iterasjon. Vi har også satt opp én Dell- og én Cisco-switch som overvåkes på labserveren. Serveren som skal stå i produksjon har også blitt installert og konfigurert med Debian og Icinga.

MonKey: Statusrapport

Onsdag 8. Mars 2013

Tredje iterasjon: Infrastruktur

I denne iterasjonen har infrastruktur vært i fokus, dette er hovedsakelig alt som skaper et sikkert miljø for applikasjonene og operere i. Våre områder i denne iterasjonen har omfattet switcher, brannmurer og UPS. Vi startet med å legge inn alle etasjeswitchene lokalt på fylkeshuset og fikk opp overvåkning for temperatur, vifter og load.

Videre ble brannmurer lagt til og parent / child konfigurering ble startet opp for å reflektere det redundante oppsettet. Dette vil være viktig for å kunne varsle riktig.

Planlegging av overvåkning for XEN og VMware har startet, og her venter vi på servicevinduet. den 13.mars for implementering av dette.

Overvåkning av MERU-kontrollerene foregår på en annen måte enn de andre enhetene. Her bruker vi SNMP traps istedenefor å polle hver enhet. Dette vil si at en enhet vil selv si ifra til icinga serveren når noe går galt. Uheldigvis støttes ikke SNMP Inform som ville gitt resending av traps dersom de ikke når fram, men vi regner ikke med at dette blir noe problem i praksis.

For alt som har blitt lagt til i overvåkningen under denne iterasjonen har vi jobbet med SNMP og brukt ulike OID-er for å hente ut ønsket informasjon. OID-ene er definert i MIB-filer som ofte er spesifikke per produsent. Uten disse MIB-filene er det umulig å vite hva det er mulig å hente ut fra enheten. Vi har derfor brukt mye tid på googling etter disse. Det har vært variert dokumentasjon og vanskelighetsgrad for å få tak i denne informasjonen, men det har gått rimelig greit.

Status etter denne iterasjonenen er at vi har fått opp et fungerende oppsett for switcher, brannmurer, kontrollere, og ups. Her inngår merkene HP, Cisco, Dell og MERU.

Et problem er å gjengi en ring topologi for switchene til fylkeshuset, men dette vil ta med videre og se nærmere på. Vi har innsett at vi ikke vil rekke å sette alt utstyret til IKT-avdelingen under overvåkning, men tar et godt utvalg som kan tjene som eksempler på hvordan resten kan implementeres. IKT-avdelingen er også inneforstått med

dette.

Ved statusmøte den 7. Mars var oppdragsgiver fornøyd.

MonKey: Statusrapport

Fredag 15. Mars 2013

Fjerde iterasjon: Varsling

I denne iterasjonen har vi satt opp varsling for service- og hostsjekker. Etter at det er blitt bestemt hvem som skal motta meldingen vil den bli formatert til å vise mest mulig relevant info. Disse blir sendt ut både til SMS og Epost. Vi har definert servicevinduene onsdag etter andre tirsdag i måneden som et unntak til varslingen.

Vi har også laget et script for synkronisering av kontakter og kontaktgrupper slik at hvem som varsles kan styres i Active Directory, og epostadresser og mobiltelefonnummer blir hentet fra brukerkontoene. Dette er satt opp og testet i lab, men det gjenstår å få det implementert i produksjon.

Vi fikk være med på service vindu 13. mars, der vi fikk lagt inn NRPE-agenter på flere Windows servere og testet at failover fungerte og ble riktig varslet på en brannmur. På servicevinduet fikk vi også satt opp round-trip-epostmonitorering, der vi sjekker at mail-sending og mottak fungerer, samt hvor lang tid det tar fra mailen er sendt til vi mottar den igjen.

Parallellt med denne iterasjonen har det også blitt satt opp overvåkning av VMware cluster og de ESX hosts som hører til der, og flere forskjellige Xen clustere med tilhørende hosts. Et lite problem med Xen var at sjekkene var veldig CPU-intensive i forhold til de andre sjekkene vi kjører. Dette er noe vi blir nødt til å se litt mer på.

I løpet av iterasjonenen hadde vi et statusmøte med Lasse, og her snakket vi om hvordan ansvarsforholdene var innad i avdelingen. Videre hadde vi en lengre gjennomgang av total status og demo av webgrensesnittet med flere fra IKT-avdelingen den 19. mars.

Etter denne iterasjonen vil vi kjøre iterasjonene Statusvindu og Servermiljø parallellt. Vi har fått utstyret som Servermiljø er avhengig av, og ser nytten i å kjøre Statusvindu nå, da det består av utvikling fra vår side. Påsken kommer i denne iterasjonsperioden og vi vil ikke jobbe samlet i denne perioden.

MonKey: Statusrapport

Fredag 29. Mars 2013

Femte iterasjon: Servermiljø

I denne iterasjonen har fokus vært på miljøet inne på serverrommet. Her har bestilt utstyr blitt montert i rack på fylkeshuset. Ledninger har blitt trukket fra overvåknings sentralen og til de fire temperatur- og luftfuktighetsfølerne. To av følerne er montert på forsiden der luften suges igjennom de rackmonterte serverne, og to bak for å se hvor mye temperaturer og luftfuktighet påvirkes av dette.

Selve sentralen er en APC netbotz 200. Denne er modulerbar og flere sensorer kan kobles til dersom man skulle ønske dette. I Icinga overvåkes nå to tjenester på sentralen, dette er luftfuktighet og temperaturer for alle følerne. Vi samler også inn performance data på dette slik at det kan lages grafer for å finne ut av trender osv for disse to måleenhetene.

Etter statusmøtet med Lasse har vi fått gode tilbakemeldinger at dette var slik de så for seg at serverrommet ble overvåket, samt at dette kommer til å bidra til å samle trender og gjøre miljøet bedre og sikkrere ved hjelp av trend data.

Siden denne iterasjonen har foregått gjennom påskeferien har vi valgt å utsette statusrapporten før hele gruppen, Svein-Inge og Lasse har hatt et statusmøte.

I denne iterasjonen skulle vi også ha lagt inn overvåkning for UPS, siden vi alt har gjort mesteparten av dette i infrastruktur modulen gjenstod det her bare å gå igjennom å kvalitetsikkre tresholds og performance data som var samlet inn.

Parallelt med denne iterasjonen vil vi jobbe med statusvisning, statusrapport for denne kommer i en egen rapport.

MonKey: Statusrapport

Fredag 12. April 2013

Sjette iterasjon: Statusvisning

I Iterasjonen statusvisning har vi laget en selvutviklet løsning for å vise status for overvåkningen på storskjerm. Vi har her skreddersydd en løsning hvor Svein-Inge og Lasse ofte har kommet med innspill på hvordan de vil ha dette, vi har her integrert

- Temperatur grafisk
- luftfuktighet grafisk
- Aktive saker på footprints
- Driftsmeldinger
- Hosts og tjeneste status fra Icinga

Vi startet med å gjøre research på hvilke løsninger som fantes. Det mest lovende vi fant var basert på Nagios med Nagios-API; nagdash. Her var utseendet og oppsettet ganske likt det vi så for oss, så vi regnet med at vi bare trengte å oversette database-kallene til kall mot Icinga-web API-et. Vi kontaktet utvikleren og ble enig om at vi kunne bruke det som utgangspunkt etter at det ble ilagt en GPL-lisens.

Det viste seg etterhvert at vi måtte skrive om mer eller mindre hele nagdash. Etter ønsker fra IKT-avdelingen har vi også lagt inn små "plugins" som viser annen relevant informasjon som statistikk over saker og RSS-feed for driftsmeldinger.

Selve utførelsen har gått gjennom flere iterasjoner der vi har satt opp en ny versjon hos IKT-avdelingen for hver gang vi har vært der, og fått tilbakemeldinger neste gang. Dette har fungert veldig godt og vi har fått avdekket et par bugs vi antagelig aldri ellers ville funnet. Vi fikk også gode inspill på fontstørrelse, farger og plasseringer. Siden har også bli testet på fargeblinde for å sjekke at fargevalgene ikke går utover lesbarheten.

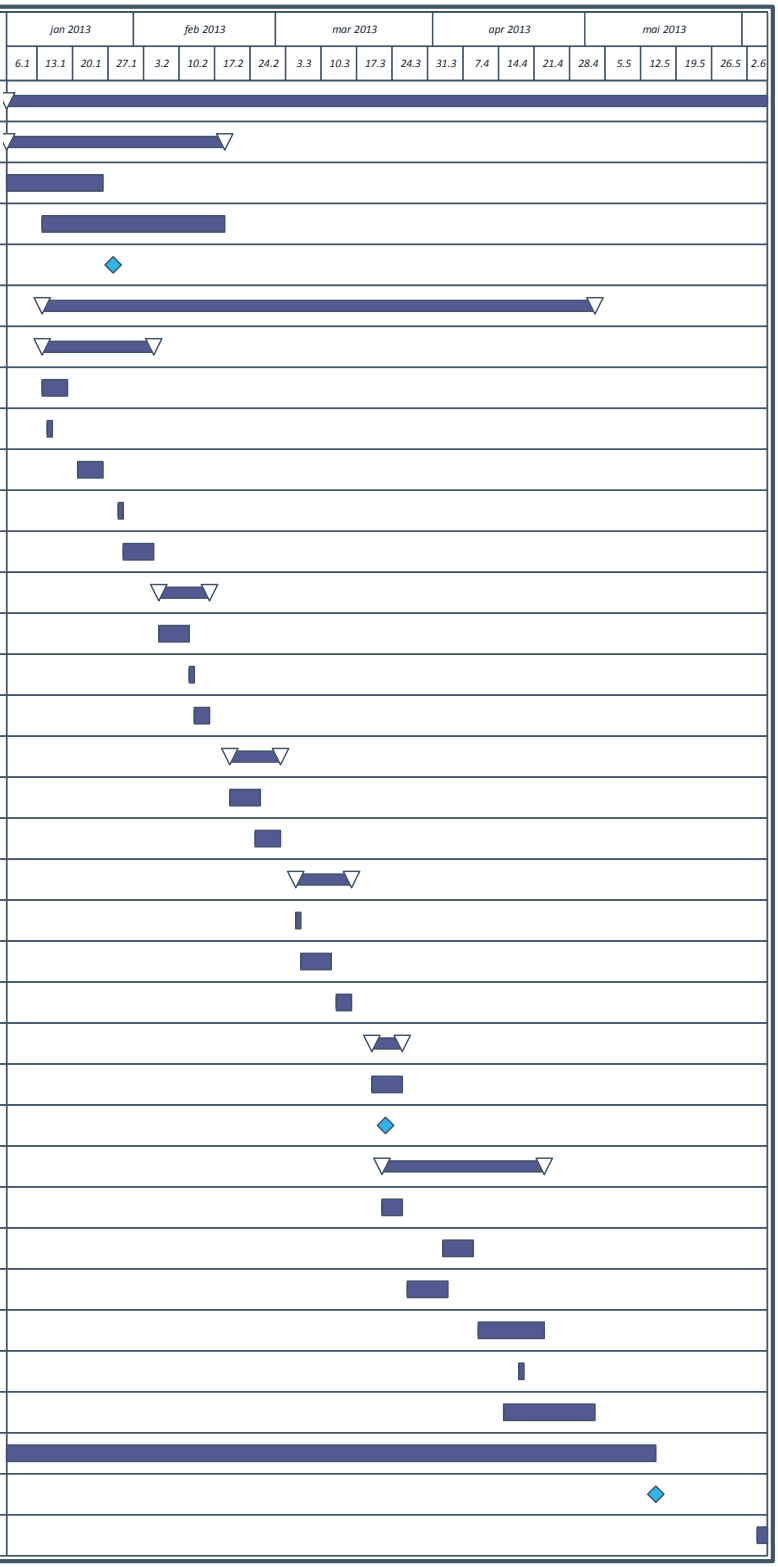
Siden vi skulle utvikle denne statusvisningen valgte vi å på forhånd ha et møte med mange av de ansatte som til daglig skal bruke denne skjermen, her fikk vi mye informasjon om hvordan dagens løsning brukes, hvordan denne nye løsningen kan bli bedre og hvordan de ønsker at den nye skal fungere.

Vedlegg F

Gantt-skjema

I dette vedlegget er Gantt-skjema for prosjektet slik det ble gjennomført.

ID	Task Name	Start	Finish	Duration
1	**Bacheloroppgave**	**07.01.2013**	**06.06.2013**	**109d**
2	**Forprosjekt**	**07.01.2013**	**18.02.2013**	**31d**
3	Utarbeide prosjektplan	07.01.2013	25.01.2013	15d
4	Opprette webside	14.01.2013	18.02.2013	26d
5	Innlevering	28.01.2013	28.01.2013	0d
6	**Implementering / Utvikling**	**14.01.2013**	**02.05.2013**	**79d**
7	**Modul Kjerneprogramvare**	**14.01.2013**	**04.02.2013**	**16d**
8	Research	14.01.2013	18.01.2013	5d
9	Møte med veileder og oppdragsgiver	15.01.2013	15.01.2013	1d
10	Oppsett og testing	21.01.2013	25.01.2013	5d
11	Fremvisning og tilbakemelding	29.01.2013	29.01.2013	1d
12	Videre oppsett og testing	30.01.2013	04.02.2013	4d
13	**Modul Server**	**06.02.2013**	**15.02.2013**	**8d**
14	Tilpassning og implementering	06.02.2013	11.02.2013	4d
15	Fremvisning og tilbakemelding	12.02.2013	12.02.2013	1d
16	Forbedring / slutt-testing	13.02.2013	15.02.2013	3d
17	**Modul Infrastruktur**	**20.02.2013**	**01.03.2013**	**8d**
18	Tilpassning og implementering	20.02.2013	25.02.2013	4d
19	Forbedring / slutt-testing	25.02.2013	01.03.2013	5d
20	**Modul Varsling**	**05.03.2013**	**15.03.2013**	**9d**
21	Fremvisning og tilbakemelding - All funksionalitet	05.03.2013	05.03.2013	1d
22	Tilpassning og implementering	06.03.2013	11.03.2013	4d
23	Forbedring / slutt-testing	13.03.2013	15.03.2013	3d
24	**Modul Servermiljø**	**20.03.2013**	**25.03.2013**	**4d**
25	Tilpassning og implementering	20.03.2013	25.03.2013	4d
26	Forbedring / slutt-testing	22.03.2013	22.03.2013	0d
27	**Modul Statusvisning**	**22.03.2013**	**22.04.2013**	**22d**
28	Research	22.03.2013	25.03.2013	2d
29	Tilpassning og implementering	03.04.2013	08.04.2013	4d
30	Påskeferie	27.03.2013	03.04.2013	6d
31	Forbedring / slutt-testing	10.04.2013	22.04.2013	9d
32	Gjennomgang av oppsett	18.04.2013	18.04.2013	1d
33	Etterarbeid	15.04.2013	02.05.2013	14d
34	Rapportskriving	07.01.2013	14.05.2013	92d
35	Innlevering av rapport	15.05.2013	15.05.2013	0d
36	Presentasjon	04.06.2013	06.06.2013	3d



Vedlegg G

Forprosjekt

I dette vedlegget finnes forprosjektet for MonKey.

HØGSKOLEN I GJØVIK

FORPROSJEKT

MonKey

"Monitoring is Key"

Av:
Øyvind Sigerstad
Nils Slåen
Bjørn-Erik Strand

Veileder:
Erik Hjelmås

28. januar 2013

Innhold

Figurer

Tabeller

1 Mål og rammer

1.1 Bakgrunn

Serviceenheten IKT ved Hedmark Fylkeskommune (heretter kalt IKT-avdelingen) har det overordnede ansvaret for alt datautstyr som tilhører fylkeskommunen og datanettverket for sentraladministrasjonen og øvrige lokasjoner. Dette omfatter blant annet videregående skoler, tannklinikker, Hedmark Trafikk m.fl. Totalt benyttes IT-løsningene av over 10 000 brukere.

Datasystemene er kritiske for daglig drift av de mange funksjonene fylkeskommunen har. Det er derfor mange brukere som berøres dersom en feil skulle inntreffe i systemene. Det er viktig at de ansvarlige får beskjed så raskt og effektivt som mulig når et system går ned eller ikke fungerer som det skal.

Per i dag benyttes et enkelt overvåkningssystem, "Servers Alive" [1], som ikke dekker IKT-avdelingens behov. Oppsettet gir for mye irrelevant informasjon, som gjør at en ikke får et godt nok oversiktsbilde. Det mangler også mulighet for å kunne overvåke mange ønskede parametere. Systemet gir kun varsling til skjerm, og det er ønskelig å også kunne få SMS og epost. Det er også en proprietær løsning og dermed ikke lett å utvide. I tillegg til dette kjøres ikke sjekkene parallelt, og derfor skalerer systemet dårlig.

IKT-avdelingen har lenge arbeidet med å finne en bedre overvåkningsløsning for infrastrukturen enn dagens. Det er ønskelig å erstatte denne løsningen med en mer modulbasert, der IKT-avdelingen har tilgang til all kildekode og kan utføre endringer.

De har imidlertid for hver runde konkludert med at de ikke har de nødvendige ressursene til å sette i gang et slikt prosjekt. Da vi forespurte IKT-avdelingen om de hadde noen mulige bachelorprosjekter, var de veldig interessert i at en bachelorgruppe kunne ta tak i denne problemstillingen.

1.2 Prosjektmål

Effektmål

I forhold til dagens løsning skal den nye løsningen:

- være mer oversiktlig.
- tilrettelegge for mer omfattende overvåkning.
- være mer fleksibel.
- bidra til mer effektiv drift for IKT-avdelingen, der hendelser raskt kan oppdages og eskaleres.

Resultatmål

- Utvikle en overvåkningsløsning som tilfredsstiller de krav som er satt.
- Den nye overvåkningsløsningen vil resultere i at den eksisterende kan erstattes og deretter fases ut.

Læringsmål

- Sette oss inn i "best practices" for overvåking av datasystemer.
- Kunne tilpasse og implementere en overvåkningsløsning for små og mellomstore bedrifter.

1.3 Rammer

Følgende krav er gitt i oppgaven:

- Dersom eksisterende programvare benyttes må denne være open-source.
- MySQL skal benyttes som databasesystem.
- Oppgavedokument skal ikke inneholde sensitiv informasjon om Hedmark fylkeskommunes tekniske løsninger. Eksempelvis IP-adresser, brukernavn/passord og brannmurkonfigurasjon. Dette er i tråd med taushetserklæringen som må inngås.
- Det skal kjøpes inn en enhet for å overvåke luftfuktighet og temperaturer. Innkjøp av denne skal skje i henhold til fylkeskommunens innkjøpsrutiner.

2 Omfang

2.1 Oppgavebeskrivelse

Det skal settes opp et overvåkningssystem som skal varsle feil og hendelser på enheter i fylkeskommunes datanettverk. Dette innebefatter servere med tjenester, og nettverksenheter som switcher, routere og brannmurer. Dette systemet må være modulært med mulighet for å enkelt kunne legge til nye moduler i ettertid. Følgende krav er satt til løsningen:

- Webgrensesnitt for visning av overvåking på skjerm.
- Varsling av flere grader kritisk nivå. Med mulighet for varsling til sms, e-post og skjerm.
- Avhengigheter mellom enheter som overvåkes skal kunne settes opp. Varsling skal gi informasjon om det er antatt følgefeil og hovedfeil.
- Systemet skal ta høyde for redundante systemer.
- Alle hendelser skal logges og lagres i database.
- Støtte WMI, SNMP, PING (med antall drop før varsling), og evt. andre aktuelle standarder.
- Det skal være enkelt å legge til nye enheter for overvåking.
- Servermiljø skal kunne overvåkes på kjøling, temperatur, luftfuktighet og UPS.

Vi må derfor finne programvare som kan tilpasses fylkeskommunes systemer og dekker flest mulig av disse kravene. Dersom eksisterende løsninger ikke finnes, eller ikke dekker behovet, må vi utvikle dette selv.

IKT-avdelingen har også en liste over moduler de ønsker implementert i løsningen. disse vil vi prøve å implementere dersom vi har tid.

- Rapporter - Uthenting av trender, dagens situasjon osv.
- Driftsarbeid - Planlagt arbeid definert mot enheter vil ikke utløse alarm men driftsarbeidsvarsling.
- SLA.
  - Vise SLA avtaler opp mot faktiske verdier.
  - Kundebase med krav opp mot logget data.
  - Kapasitetsmålinger opp mot maksverdier (f.eks. linjekapasitet).
- Dashboard funksjonalitet med mulighet for personliggjøring av eget grafisk grensesnitt.
- Eget grensesnitt for mobile enheter.
- Konfigurasjonskart - Tegner kart av løsning grafisk med klikkbare objekter.
- CMDB - Mulighet for lagring av konfigurasjonsoppsett av objekter (enheter).
- Telefoni - Overvåking av telefonlinjer opp mot Trio og Asterisk.

2.2 Avgrensninger

Siden vi har både må-krav og ønsket-krav til funksjonalitet, må vi avgrense oss slik at vi overholder tidsrammer i forhold til må-krav. Om vi ser at vi ligger foran tidsskjema og har fått på plass alle må-kravene til implementasjonen, vil vi begynne å se på ønsket funksjonalitet.

3 Prosjektorganisering

3.1 Roller

Prosjektleder Prosjektleder vil være Øyvind Sigerstad. Denne rollen vil være fast under hele prosjektet. Lederens hovedansvar er å se nødvendigheten for møter, samt ha et overordnet ansvar for arbeidsfordelingen. Prosjektlederen vil også ta endelige avgjørelser dersom gruppen ikke kan komme til enighet, som stipulert i gruppereglene i Vedlegg A.

Kommunikasjonsansvarlig Vår kontaktperson er Nils Slåen. All kontakt med oppdragsgiver og annen utgående kommunikasjon vil gå via ham. Kommunikasjonsansvarlig har også ansvar for å avtale møter.

Webansvarlig Vår webansvarlig er Bjørn-Erik Strand. Hans hovedansvar er å sette websiden vår i drift før fristen. Webansvarlig har også et overordnet ansvar for å få oppdateringer publisert. I tillegg til at gruppens medlemmer skal ha tilgang til å utføre oppdateringer.

Oppdragsgiver Vår oppdragsgiver er Svein-Inge Kvalø (Fungerende driftsleder ved IKT-Avdelingen, Hedmark fylkeskommune). Han er vårt kontaktpunkt ved Hedmark fylkeskommune. Lasse Odden er vår tekniske kontakt (IT-konsulent ved IKT-Avdelingen - Hedmark fylkeskommune). Han vil være en ressurs vi kan bruke om nødvendig, oppdragsgiver skal være informert om dette.

Veileder Erik Hjelmås (Førsteamanuensis, Dr. scient ved HiG), er vår veileder gjennom bacheloroppgaven.

3.2 Rutiner og regler i gruppa

Et eget dokument er utarbeidet med grupperegler som alle gruppens medlemmer er enige i og har skrevet under på. Dette finnes under som Vedlegg A.

3.3 Verktøy

Alt av egenutviklet kildekode og konfigurasjonsfiler vil bli lagt under et SVN repository. Dette for å få versjonskontroll og samle alt på et sted, som vi kan ta backup av. Google Docs vil bli brukt for å enkelt kunne samarbeide på dokumenter. Når den endelige rapporten skal genereres, vil vi importere Google Docs dokumentet til LaTeX. Dropbox vil bli benyttet til å dele filer og dokumenter innad i prosjektgruppen, og med oppdragsgiver. Vi vil bruke Wordpress som publiseringsverktøy for hjemmesiden.

4 Planlegging, oppfølging og rapportering

4.1 Utviklingsmodell/rammeverk

For å tilpasse overvåkningssystemet til IKT-avdelingens krav, vil det med stor sannsynlighet komme endringer etterhvert. Det vil kunne oppstå ulike situasjoner som er vanskelige å forutse. En plugin som kreves eksiterer ikke, webgrensesnittet må endres på grunn av mer informasjon eller omorganisering, eller oppdragsgiver har innspill som fører til endring av funksjonalitet i ettertid. Dette fører til at vi må ha en fleksibel modell med tanke på endringer og tilbakefall, men med begrensninger på hvor mye tid vi kan bruke ved uforutsette problemer.

På grunn av en naturlig sammenheng og avhengigheter mellom kravene som er satt, har vi valgt å gruppere funksjonalitet i moduler. De forskjellige modulene er "Kjerneprogramvare", "Server", "Infrastruktur", "Servermiljø", "Varsling" og "Statusvindu" (visualisert i Figur 2). Med nærliggende sammenheng mener vi at funksjonalitet som CPU, RAM, disk og prosesser er i samme kategori. Planen er også å gjennomføre disse i en gitt rekkefølge. Varsling kan settes opp uten noe input, men med tanke på redundans i systemene, avhengigheter mellom objektene, og testing av funksjonene som går inn under varsling ser vi på dette som en avhengighet av "Server" og "Infrastruktur". "Statusvindu" har vi valgt å legge sist fordi vi ser for oss at modulene "Server", "Infrastruktur, "Servermiljø", og "Varsling" må være på plass før en varslingsskjerm kan ferdigstilles.

I tradisjonell systemutvikling må man definere alle parametere og rekkefølge på alt som skal utvikles, før en starter med selve utviklingen. En smidig modell vil gi oss mer frihet til å gjøre justeringer underveis. Vi har derfor valgt en iterativ modell [2]. En del av den iterative modellen er "product control list", men denne har vi valgt å ikke ta med fordi vi har satt opp en overordnet kategorisering av funksjonalitet, og bestemt når hver av disse skal implementeres.

Vi vil dele inn prosjektet i faser, som hver består av en planleggingsfase, en implementeringsfase, og en fase for testing og evaluering. Disse vil så gjentas for hver modul som skal implementeres. Halvveis i hver iterasjon vil vi ha et møte med oppdragsgiver der vi kan presentere funksjonalitet i modulen, og få tilbakemeldinger. Det gjør at vi kan jobbe med dette i den siste halvdelen av iterasjonen. Dette vil forhåpentligvis føre til at produktet ikke sporer av fra de forventningene og behovene IKT-avdelingen har.

4.2 Labutstyr

Prosjektet omfatter mye testing mot IKT-avdelingens systemer. Vi har derfor bedt om å få vårt eget labmiljø hvor vi kan utføre lab-scenarier uten å påvirke den daglige driften til IKT-avdelingen. Dette miljøet vil i første omgang inneholde to Linux-servere, og en Windows-server. Her vil vi også trenge muligheten til å koble inn fysisk utstyr, som for eksempel brannmurer og switcher. Dette løses ved å bruke et eget VLAN som tagges ut til fysiske porter. Serverene i seg selv kan gjerne være virtualisert i for eksempel VMware. Det vil bli naturlig å utvide dette labmiljøet etterhvert, når vi skal se på overvåkning av forskjellige tjenester.

4.3 Møter

Vi har lagt opp til et ukentlig statusmøte med veileder Erik Hjelmås. Som tidligere nevnt, har vi også planlagt å ha et statusmøte annenhver uke med oppdragsgiver. Disse vil inneholde korte fremgangsrapporter og en mulighet for IKT-avdelingen til å komme med innspill på det som er gjort i iterasjonen. Da vil de også kunne komme med eventuelle ønsker og forslag til endringer. Vi vil også ha løpende kontakt med vår tekniske kontakt ved IKT-avdelingen.

5 Organisering og kvalitetskontroll

5.1 Dokumentasjon standardbruk og kildekode

Etter hvert som konfigurasjon og funksjonalitet blir implementert, må også dokumentasjon skrives og følges opp. Dokumentasjonen skal følge den standarden fylkeskommunen har for eksisterende løsninger.

Alle konfigurasjonsfiler skal kommenteres med hovedfunksjon og dato endret øverst. Videre skal hver linje kommenteres in-line der det er hensiktsmessig. Kildekode vi skriver skal følge den standarden som er brukt i programvaren vi baserer løsningen på.

Beskrielse	Sannsynlighet	Konsekvens	Risiko
Tap av data	1	8	8%
At programvaren vi velger å basere løsningen på ikke er god nok	4	8	32%
Utvikling/implementering av funksjon blir for omfattende	5	5	25%
Overskridelse av tidsrammer	3	7	21%
Løsningen tilfredstiller ikke oppdragsgivers krav	3	7	21 %
Lengre fravær grunnet sykdom	1	7	7%

Tabell 1: Risikoanalyse

5.2 Risikoanalyse

I Tabell 1 har vi listet opp relevante risikoer for prosjektet.

Tiltak for å redusere kritiske risikoer At programvaren vi velger å basere løsningen på ikke er god nok, vil være en signifikant risiko fordi vi kan bli nødt til å finne ny programvare å basere løsningen på, og må begynne fra bunnen av. Derfor er det viktig at vi på forhånd har gjort nødvendig research før vi velger en kjerneprogramvare å benytte. Ved planlagte møter med oppdragsgiver vil vi også kunne avdekke hvor mye tilpasning som skal til for å få til en løsning de er fornøyde med.

For å unngå at en funksjon i modulene tar for lang tid, vil vi i første fase av hver iterasjon dele modulen inn i mindre oppgaver. Da vil vi skaffe oversikt over hva som må gjøres og gjenstår. Deretter vil vi fordele ressurser etter oppgavenes prioritet og hvor lang tid vi tror de vil ta.

Dersom en funksjon tar lengre tid enn planlagt og er kritisk for prosjektet, vil dette kunne føre til at prosjektets tidsrammer forskyves. Dette kan gå ut over andre funksjoner eller moduler. Her er kontakt med oppdragsgiver viktig slik at de kan være med på å avgjøre hva som skal gjøres.

Om vår løsning ikke skulle møte oppdragsgivers krav vil dette påvirke prosjektets helhet. Dette forebygges ved å ha hyppige møter med både oppdragsgiver og teknisk kontakt. På denne måten har vi mulighet til å foreta små justeringer underveis i prosjektet.

6 Plan for gjennomføring

6.1 Milepæler

- Innlevering av forprosjekt 27.01.2013
- Webside skal være oppe 20.02.2013
- Innlevering av rapport 15.05.2013

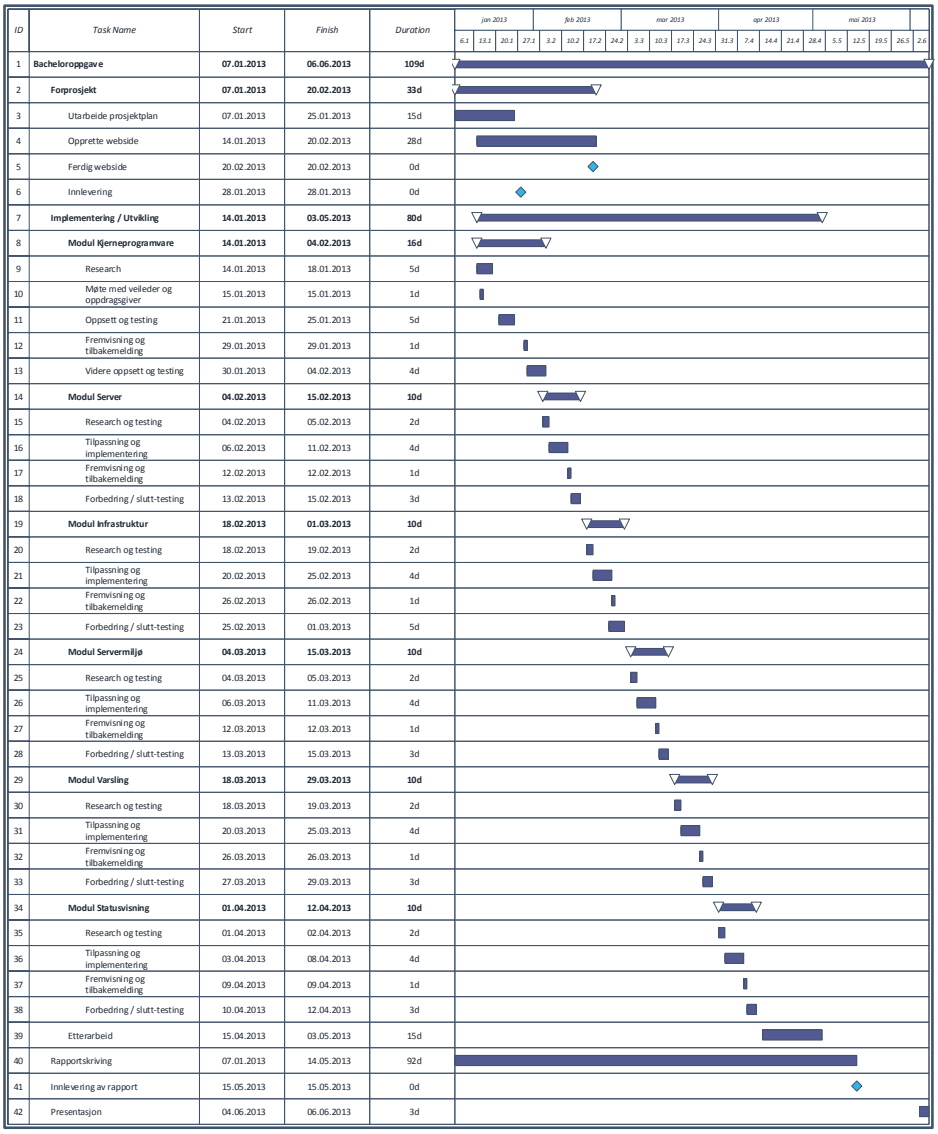
6.2 Gantt-skjema

I Figur 1 har vi satt hver iterasjon til å vare i 2 uker (10 arbeidsdager). Vi ser for oss at alle iterasjoner går gjennom fasene som er listet. Men vi regner med at vi ikke vil bruke like lang tid på hver fase i alle modulene. Vi har satt opp den rekkefølgen for gjennomføring av modulene vi mener er naturlig, men ser også at det kan bli nødvendig å endre på denne underveis.

Dersom "Server" og "Infrastruktur" overskrider sine rammer, vil vi nedprioritere modulen "Servermiljø". Da vil vi gå videre på "Varsling", og "Servermiljø" vil bli vurdert gjenopptatt ved avsatt tid som "etterarbeid", eller tid til overs fra "Varsling" og "Statusvindu". Det kan også bli aktuelt å implementere modulene "Server" og "Infrastruktur" parallelt, men vi har satt av tid til begge individuelt i utgangspunktet.

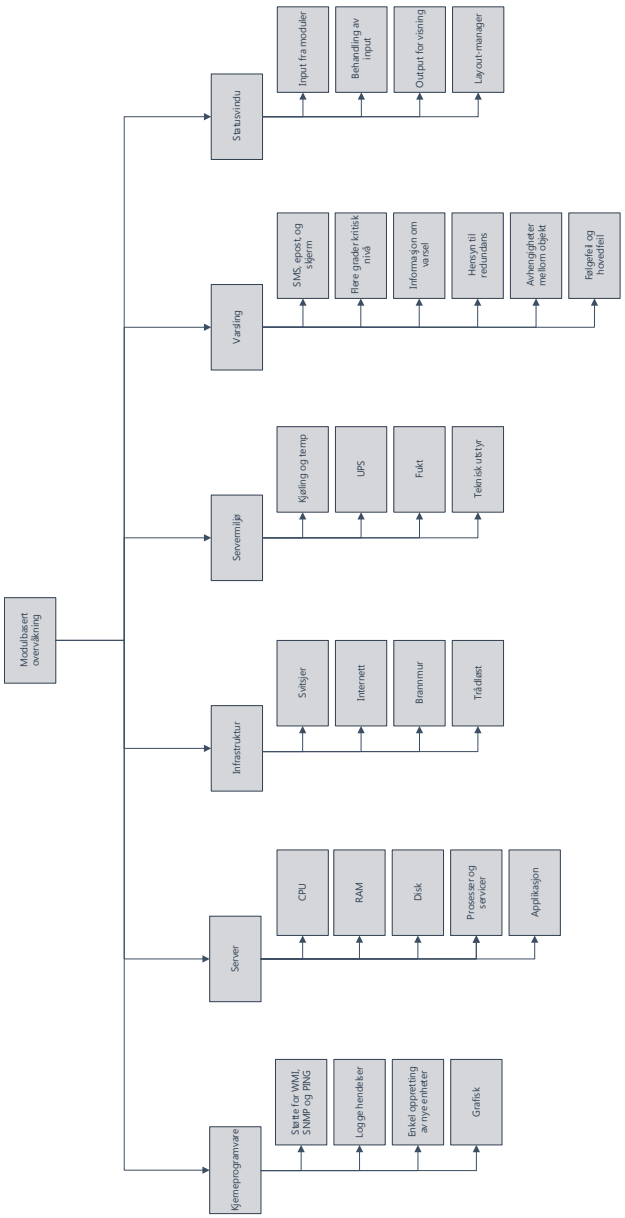
6.3 WBS

I Figur 2 ser vi en visuell framstilling av de ulike modulene, og hvilke underliggende komponenter hver modul inneholder.



ID	Task Name	Start	Finish	Duration
1	**Bacheloroppgave**	**07.01.2013**	**06.06.2013**	**109d**
2	**Forprosjekt**	**07.01.2013**	**20.02.2013**	**33d**
3	Utarbeide prosjektplan	07.01.2013	25.01.2013	15d
4	Opprette webside	14.01.2013	20.02.2013	28d
5	Ferdig webside	20.02.2013	20.02.2013	0d
6	Innlevering	28.01.2013	28.01.2013	0d
7	**Implementering / Utvikling**	**14.01.2013**	**03.05.2013**	**80d**
8	**Modul Kjerneprogramvare**	**14.01.2013**	**04.02.2013**	**16d**
9	Research	14.01.2013	18.01.2013	5d
10	Møte med veileder og oppdragsgiver	15.01.2013	15.01.2013	1d
11	Oppsett og testing	21.01.2013	25.01.2013	5d
12	Fremvisning og tilbakemelding	29.01.2013	29.01.2013	1d
13	Videre oppsett og testing	30.01.2013	04.02.2013	4d
14	**Modul Server**	**04.02.2013**	**15.02.2013**	**10d**
15	Research og testing	04.02.2013	05.02.2013	2d
16	Tilpassning og implementering	06.02.2013	11.02.2013	4d
17	Fremvisning og tilbakemelding	12.02.2013	12.02.2013	1d
18	Forbedring / slutt-testing	13.02.2013	15.02.2013	3d
19	**Modul Infrastruktur**	**18.02.2013**	**01.03.2013**	**10d**
20	Research og testing	18.02.2013	19.02.2013	2d
21	Tilpassning og implementering	20.02.2013	25.02.2013	4d
22	Fremvisning og tilbakemelding	26.02.2013	26.02.2013	1d
23	Forbedring / slutt-testing	25.02.2013	01.03.2013	5d
24	**Modul Servermiljø**	**04.03.2013**	**15.03.2013**	**10d**
25	Research og testing	04.03.2013	05.03.2013	2d
26	Tilpassning og implementering	06.03.2013	11.03.2013	4d
27	Fremvisning og tilbakemelding	12.03.2013	12.03.2013	1d
28	Forbedring / slutt-testing	13.03.2013	15.03.2013	3d
29	**Modul Varsling**	**18.03.2013**	**29.03.2013**	**10d**
30	Research og testing	18.03.2013	19.03.2013	2d
31	Tilpassning og implementering	20.03.2013	25.03.2013	4d
32	Fremvisning og tilbakemelding	26.03.2013	26.03.2013	1d
33	Forbedring / slutt-testing	27.03.2013	29.03.2013	3d
34	**Modul Statusvisning**	**01.04.2013**	**12.04.2013**	**10d**
35	Research og testing	01.04.2013	02.04.2013	2d
36	Tilpassning og implementering	03.04.2013	08.04.2013	4d
37	Fremvisning og tilbakemelding	09.04.2013	09.04.2013	1d
38	Forbedring / slutt-testing	10.04.2013	12.04.2013	3d
39	Etterarbeid	15.04.2013	03.05.2013	15d
40	Rapportskriving	07.01.2013	14.05.2013	92d
41	Innlevering av rapport	15.05.2013	15.05.2013	0d
42	Presentasjon	04.06.2013	06.06.2013	3d

Figur 1: Gantt-skjema



Figur 2: Work breakdown structure

Referanser

[1] Woodstone bvba. Servers alive hjemmeside. http://www.woodstone.nu/salive/, 2013. [Online; accessed 15-January-2013].

[2] Wikipedia. Iterative and incremental development — wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Iterative_and_incremental_development&oldid=532022769, 2013. [Online; accessed 20-January-2013].

Grupperegler for MonKey

1. Arbeidsregler
Hvert medlem skal minimum jobbe 30 timer i uka på prosjektet. Gruppen vil føre arbeidslogg i fellesskap. Gruppens arbeidstid er hverdager mellom 08:00 og 16:00, med unntak av forelesninger som følges av hvert enkelt gruppemedlem. Ved fravær skal det gis beskjed til gruppeleder med begrunnelse. Arbeidsoppgaver bestemmes og utdeles i fellesskap. Dersom et medlem ikke utfører avtalt arbeid, skal dette tas opp sammen med prosjektleder.

2. Prosjektleder
Øyvind Sigerstad vil fungere som prosjektleder gjennom hele prosjektet. Prosjektleder representerer gruppen utad og har fullmakt til å signere dokumenter på vegne av hele gruppen.

3. Beslutninger
Beslutninger avgjøres ved avstemning. Dersom en avgjørelse ikke får flertall, vil en ny avsteming finne sted etter en kort periode med informasjonsinnhenting rundt problemstillingen. Er det fortsatt ikke flertall vil prosjektleders stemme telle dobbelt.

4. Kostnader
Alle kostnader gruppen har i forbindelse med prosjektet vil bli fordelt likt på alle gruppemeldemmer.

5. Regelbrudd
Ved gjentagende brudd på regler, vil saken bli tatt opp med veileder. Veileder vil i samarbeid med gruppens medlemmer bestemme utfallet.

Øyvind Sigerstad — 28.01.2013

Nils Slåen — 28.01.2013

Bjørn-Erik Strand — 28.01.2013

HØGSKOLEN I GJØVIK

PROSJEKTAVTALE

mellom Høgskolen i Gjøvik (HiG) (utdanningsinstitusjon),

Hedmark fylkeskommune, Serviceenheten IKT

(oppdragsgiver), og

Øyvind Sigerstad,
Nils Slåen,
Bjørn-Erik Strand

(student(er))

Avtalen angir avtalepartenes plikter vedrørende gjennomføring av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer:

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 07.01.2013 til 15.05.2013.

   Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning. Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider. Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet. Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper. Oppdragsgiver plikter på forespørsel fra HiG å gi en vurdering av prosjektet vederlagsfritt.

2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
   - Oppdragsgiver dekker selv gjennomføring av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser og nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstillelse av den skriftlige besvarelsen vedrørende prosjektet.
   - Eiendomsretten til eventuell prototyp tilfaller den som har betalt komponenter og materiell mv. som er brukt til prototypen. Dersom det er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.

3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.

4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter, taper mv. som inngår som del av eller vedlegg til besvarelsen, gis det en kopi av til HiG, som vederlagsfritt kan benyttes til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.

   Besvarelser med karakter C eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på bibliotekets del av skolens internett-sider. Dette avhenger av at studentene skriver under på en egen avtale hvor de gir biblioteket tillatelse til at deres hovedprosjekt blir gjort tilgjengelig i papir og nettutgave (jfr. Lov om opphavsrett). Oppdragsgiver og veileder godtar slik

offentliggjøring når de signerer denne prosjektavtalen, og må evt. gi skriftlig melding til studenter og dekan om de i løpet av prosjektet endrer syn på slik offentliggjøring.

5. Besvarelsens spesifikasjoner og resultat kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten(e) i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student(er) bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.

6. Ut over den offentliggjøring som er nevnt i punkt 4 har studenten(e) ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student(er) og faglærer/veileder for det materialet som faglærer/veileder stiller til disposisjon.

7. Studenten(e) leverer oppgavebesvarelsen med vedlegg (pdf) i Fronter. I tillegg leveres et eksemplar til oppdragsgiver.

8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det dekan/prodekan som godkjenner avtalen.

9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere regulerer forhold vedrørende bl.a. eiendomsrett, videre bruk, konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.

   Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.

10. Når HiG også opptrer som oppdragsgiver trer HiG inn i kontrakten både som utdanningsinstitusjon og som oppdragsgiver.

11. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kapittel 32.

12. Deltakende personer ved prosjektgjennomføringen:

HiGs veileder (navn): Erik Hjelmås

Oppdragsgivers kontaktperson (navn): Svein-Inge Kvalø

Student(er) (signatur): Øyvind Sigvaldsen dato 15/1-2013

Nils Claes dato 15.01.2013

Bjørn-Erik Strand dato 15.01.2013

_____ dato _____

Oppdragsgiver (signatur): Svein-Inge Kvalø dato 15/1-2013

IMT Dekan/prodekan (signatur): _____ dato _____

Vedlegg H

Kildekode

Selvprodusert kildekode er lagt ved under dette vedlegget. Innholder plugin-er, script for generering av host og kontakter, init-script, SQL brukeroppretting (ikke skrevet selv) og differansen på endrede plugin-er.

H.1 Host-generering

script/gen.bash

```
#!/bin/bash
#
# This script is used to generate hosts for Icinga/Nagios
#   its NOT ment to replace manual labour its merly a tool
#   to simplify the adding of multiple hosts, from a common file format.
#
# MonKey 2013
#
#
## USAGE
# Run this script and generate hosts from a supplyed CSV file.
# The first argument is the location of the CSV file, this needs to be supplyed
#   The CSV file needs to columns at minimum, the first has to be hostname
#     the secound column has to be the IP Address
#   The third and forth columns are optional. and the default value can be changed.
#     the third column is hostgroup
#     the fourth column is the generic
# The secound argument is the path to where the files are to be generated.
#   If this is not supplyed, the hosts will be generated in your working directory.
#
#
## CSV Format:
# hostname,address,hostgroup,generic
# dc1,10.0.0.1,windows_servers,generic_service
# dc2,10.0.0.2,,generic_service
# mail1,10.0.0.3
# print1,10.0.0.4,print_servers
# apache1,10.0.0.5,windows_servers;print_servers;terminal_servers
#
#
## Arguments
# $1 = Path to csv file
# $2 = Path to generate hosts

# Default value variables

defaultGeneric="generic_ts"      # Default, not used if supplyed
defaultHostgroup="windows_servers"  # Default, not used if supplyed


```

```
41 ##############################################
42 ## Nothing should be changed below this line  ##
43 ##############################################
44
45 if [ -z "$1" ]; then        # Check if CSV file is supplyed
46   echo -e "\tThis script generates config files for Icinga/Nagios"
47   echo -e "\t Supply a CSV file with format hostname,address (See script for more options)"
48     echo -e "\tFirst argument has to be the CSV file"
49   echo -e "\tSecound argument is optional and can contain the path to where fil"
50     exit              # Terminate script
51 fi
52
53 NROFHOSTS=0             # Total number of hosts (lines) in CSV file
54 NROFERRORS=0            # Nr of hosts in CSV file without address or hostname
55 OLDIFS=$IFS             # Store away seperator
56 IFS=","                 # Seperator for CSV file
57
58 if [ -z "$2" ]; then        # Where are files created message
59   echo -e "\n\tGenerating hosts from $1 into working directory"
60 else
61   echo -e "\n\tGenerating hosts from $1 into directory $2"
62 fi
63
64 while read address hostname hostgroup generic # Read CSV
65   do
66     let "NROFHOSTS += 1"    # Counts times in the loop
67     if [ -z "$hostname" ]; then # If there is no hostname defined
68       if [ $NROFERRORS -lt 1 ]; then
69         echo -e "\n\tHostname and Address are mandatory."
70       fi
71       echo -e "\tMissing for host on line nr: $NROFHOSTS"
72       let "NROFERRORS += 1" # Counts how many errors
73
74     else            # Else there is a hostname defined
75       file=""         # Filename
76       if [ -z $2 ]; then    # Add path if supplyed
77         file="$2"
78       fi
79
80       file+="$hostname"   # Hostname will be name of the file
81       file+=".cfg"
82
83       ## Echo to file starts ##
84       echo "define host {" > "$file"
85
86       if [ -z "$generic" ]; then  # Use default if no generic defined
87         echo -e "\tuse\t\t\t$defaultGeneric" >> "$file"
88       else          # Else use supplyed
89         echo -e "\tuse\t\t\t$generic" >> "$file"
90       fi
91
92       echo -e "\taddress\t\t$address" >> "$file"
93       echo -e "\thost_name\t$hostname" >> "$file"
94       echo -e "\talias\t\t$hostname" >> "$file"
95
96       if [ -z "$hostgroup" ]; then  # Use default if no hostgroup supplyed
97         echo -e "\thostgroups\t$defaultHostgroup" >> "$file"
98       else          # Else use supplyed
99         hostgroups=${hostgroup//;/,}  # Replace ; with , for icinga config
100        echo -e "\thostgroups\t$hostgroups" >> "$file"
101      fi
102
103      echo "}" >> "$file"
104      ## Echo to file ends ##
105    fi
106  done < $1
107
108 echo -e "\n\tDone"
109 echo -e "\tSuccessfully created "$((NROFHOSTS - NROFERRORS))" hosts"
110 echo -e "\t$NROFERRORS hosts where not create because of errors."
111
112 IFS=$OLDIFS           # Set seperator back to old
113 exit              # Terminate script
```

H.2 Synkronisering av kontakter fra Active Directory

script/ad_sync.pl

```
#! /usr/bin/env perl
# Script to sync Active Directory entries to Icinga contacts
#
# Note maxValRange by default is 5k in 2k8, 10k in 2k3, meaning the max number of entries returned
# this is OK for our implementation
#
# If the option gen_service is set a generic service for each contact group is generated, but only
# if it does not already exist.
#
# MonKey 2013

use warnings;
use strict;

use Getopt::Long;
use Net::LDAP;
use File::Copy;
use constant {
    LDAP_URL => "host.example.org",
    LDAP_PORT => 3268,
    LDAP_SCHEME => "ldap",
    BIND_DN => "CN=icinga,OU=Serviceaccounts,DC=example,DC=org",
    BIND_PASSWD => "icinga_pwd",
    BASE_USER_DN => "OU=Users,DC=example,DC=org",
    BASE_GROUP_DN => "OU=icinga_contacts,DC=example,DC=org",
    ALL_CONTACTS_GROUP => "CN=all_icinga_contacts,OU=icinga_contacts,DC=example,DC=org",
};

my $gen_service = 0;
my $exit_status = 0;
my $ldap = Net::LDAP->new(LDAP_URL, port => LDAP_PORT, scheme => LDAP_SCHEME) or die "Unable to connect to ldap server $@";
my $bind = $ldap->bind(BIND_DN, password => BIND_PASSWD);

# Find out if we should generate generic service with the contactgroups
GetOptions("gen_service" => \$gen_service);

# If we can't bind to AD, theres no point in going further
if($bind->code != 0) {
  die "Unable to bind to LDAP: " . $bind->error_desc;
}

# First we need to get all contacts
my @contact_entries = group_members(ALL_CONTACTS_GROUP);
my @contacts;

# Write the contacts
foreach my $entry (@contact_entries) {
  my %contact = contact($entry);
  push(@contacts, %contact);
  write_contact(%contact);
}

# Now we can do the groups. Query for all of them
my $groups = $ldap->search(filter => "objectClass=group",
          base   => BASE_GROUP_DN,
                 attrs  => "cn");
$groups->code && die $groups->error;

# Write all groups
foreach my $group($groups->entries) {
  my %contactgroup = contactgroup($group);
  write_contactgroup(%contactgroup);
  if($gen_service) {
    write_gen_service(%contactgroup);
  }
}

```

```
68 # Done. Exit with code 2 if we have warnings
69 exit $exit_status;
70
71 # Gets all group members of a group with passed DN recursively, i.e., descend into children
       groups
72 # Returns an array of NET::LDAP::Entry objects
73 sub group_members {
74   my $DN = $_[0];
75   my $results = $ldap->search(filter => "(&(memberOf:1.2.840.113556.1.4.1941:=$DN)(objectClass=
         user)(objectCategory=person))",
76                               base   => BASE_USER_DN,
77                                         attrs  => "mail", "sAMAccountName");
78   #print $results->count();
79   $results->code && die $results->error;
80   return $results->entries;
81 }
82
83 # Gets relevant information for a contact from a passed NET::LDAP::Entry object
84 # Returns an asociative array
85 sub contact {
86   my $entry = $_[0];
87   my %contact;
88
89   $contact{'sAMAccountName'} = $entry->get_value('sAMAccountName');
90   $contact{'cn'} = $entry->get_value('cn');
91   $contact{'mobile'} = $entry->get_value('mobile');
92   $contact{'mail'} = $entry->get_value('mail');
93
94   if(!defined $contact{'mobile'}) {
95     warn "Mobile is not set for $contact{'cn'}";
96     $contact{'mobile'} = '';
97     $exit_status = 2 if $exit_status == 0;
98   }
99   if(!defined $contact{'mail'}) {
100    warn "Mail is not set for $contact{'cn'}";
101    $contact{'mail'} = '';
102    $exit_status = 2 if $exit_status == 0;
103  }
104
105  return %contact;
106 }
107
108 # Gets relevant information for a contactgroup from a passwd NET::LDAP::Entry object
109 # Returns an asociative array
110 sub contactgroup {
111   my $group = $_[0];
112   my @member_list = group_members($group->get_value('distinguishedName'));
113
114   my @members;
115   my %contactgroup;
116
117   foreach my $member(@member_list) {
118     push(@members, $member->get_value('sAMAccountName'));
119   }
120
121   $contactgroup{'members'} = join(',', @members);
122   $contactgroup{'sAMAccountName'} = $group->get_value('cn');
123   $contactgroup{'cn'} = $group->get_value('cn');
124
125   return %contactgroup;
126 }
127
128 # Writes a contact to file
129 # Takes an associative array with information as parameter
130 sub write_contact {
131   my %contact = @_;
132   my $file_name = "/etc/icinga/objects/contacts/$contact{'sAMAccountName'}_contact.cfg";
133
134   my $data =  "define contact {
135     contact_name    $contact{'sAMAccountName'}    ; AD account
136     use             generic_contact
137     alias           $contact{'cn'}    ; Full name of user
138     email           $contact{'mail'}
139     pager   $contact{'mobile'}
140   }";
```

```
141
142     write_config_file($file_name, $data);
143 }
144
145 # Writes a contactgroup to file
146 # Takes an associative array with information as parameter
147 sub write_contactgroup {
148   my %contactgroup = @_;
149   my $file_name = "/etc/icinga/objects/contactgroups/$contactgroup{'sAMAccountName'}
          _contactgroup.cfg";
150
151   my $data = "define contactgroup {
152     contactgroup_name $contactgroup{'sAMAccountName'}
153     alias       $contactgroup{'cn'}
154     members       $contactgroup{'members'}
155   }";
156
157   write_config_file($file_name, $data);
158 }
159
160 # Writes a generic service for a contact group if one does not already exist
161 sub write_gen_service {
162   my %contactgroup = @_;
163   my $file_name = "/etc/icinga/objects/generics/services/$contactgroup{'sAMAccountName'}.cfg";
164
165   unless (-e $file_name) {
166     my $data = "# Generated generic from sync. This file will not be overwritten and can be
          used as a template for services where you want this contactgroup notified.
167     define service {
168       use         generic_service
169       name              $contactgroup{'sAMAccountName'}
170       contact_groups    $contactgroup{'sAMAccountName'}
171       register                    0
172     }";
173     open(my $fh, '>', $file_name) or die "Could not open file $file_name: $!";
174       print $fh $data;
175       close $fh;
176   }
177 }
178
179 # Writes a config file
180 # Checks if the config is valid, and rolls back if a config file with the same name existed
      before
181 # Note that no files will be written if there's already an error with the icinga config
182 #
183 # Parameters are:
184 # 0: The full path to the file to be written
185 # 1: The data to be written to the file
186 sub write_config_file {
187   my $file_name = $_[0];
188   my $data = $_[1];
189   my $rollback = 0;
190
191   # If config file already exists, take a backup so we can roll back on error
192   if(-e $file_name) {
193     copy($file_name, "${file_name}_bak") or die "Could not copy $file_name for backup: $!";
194     $rollback = 1;
195   }
196
197   # Write new config file
198   open(my $fh, '>', $file_name) or die "Could not open file $file_name: $!";
199   print $fh $data;
200   close $fh;
201
202   # Check config
203   my @check = `icinga -v "/etc/icinga/icinga.cfg"`;
204
205   # Exit status was not success
206   if($? != 0) {
207     unlink $file_name or die "Could not delete $file_name: $!"; # Remove our config file
208     print "\nError in processing contact $file_name: \n";
209     for(@check) {
210       $exit_status = 1;
211       print if(/^Error:/);
212     }
```

```
213
214     # Copy back our backup if we have one
215     if($rollback) {
216       copy("${file_name}_bak", $file_name) or die "Could not restore backup file $file_name: $!
              ";
217       print "Restored previous version of $file_name \n";
218     }
219   }
220
221   # Remove our backup file
222   if($rollback) {
223     unlink("${file_name}_bak") or die "Could not delete backup file $file_name: $!";
224   }
225 }
```

H.3 Check NDB Mem

script/check_ndb_mem.pl

```
1  #!/usr/bin/env perl
2  # check_ndb_mem.pl
3  # checks memory usage on specified nodes
4  #
5  # MonKey, 2013
6
7  use strict;
8  use warnings;
9
10 use Getopt::Long;
11
12 my $check_command = '/usr/sbin/ndb_mgm';
13 my $check_expression = 'report MemoryUsage';
14 my $warn_thresh=70;
15 my $crit_thresh=80;
16
17 my $host=undef;
18 my $nodes=undef;
19 my $usage=undef;
20 my $debug=undef;
21 my $exit_code=0;
22
23 GetOptions(
24       "host=s"       => \$host,
25       "nodes=s"    => \$nodes,
26       "warning=s"    => \$warn_thresh,
27       "critical=s"   => \$crit_thresh,
28       "usage"      => \$usage,
29       "debug"      => \$debug,
30 );
31
32 sub main {
33   if ($usage || !defined $host || !defined $nodes ) {
34     usage();
35     exit(3); # Exit with code for UNKNOWN
36   }
37
38   my @node_ids = split(",", $nodes);  # Get the comma separated IDs from the option
39   my $status = "";
40
41   foreach(@node_ids) { # Loop all IDs given
42     if($_ !~ /^\d+$/) {
43       print "Non-numeric node id, " . $nodes . " is not valid \n";
44       exit(3); # Exit with code for UNKNOWN
45     }
46
47     $status .= check_mem($_);
48   }
49
50   # Give CRITICAL if one or more of the nodes have exceeded threshold
51   if($exit_code == 2) {
```

```
52      print "CRITICAL memory threshold exceeded | $status\n";
53    }
54    elsif($exit_code == 1) {
55      print "WARNING memory threshold exeeded | $status\n";
56    }
57    else {
58      print "OK | $status\n";
59    }
60
61    exit($exit_code);
62  }
63
64  # Checks memory usage for the node id passed as argument
65  # Sets the global is_critical or is_warning if applicable
66  sub check_mem {
67    my ($node) = @_;
68    my $data;
69    my $index;
70
71    my $command = "$check_command $host -e \"$node $check_expression\" 2>&1";
72    my @ndb_output = `$command`; # Get output from ndb_mgm and STDERR
73    if($? != 0) {
74      print "Error running ndb command: " . $command . "\n";
75      exit(3); # Exit with code for UKNOWN
76    }
77
78    # Loop output and grab percent used
79    foreach(@ndb_output) {
80      if ($_ =~ m/Data/) {
81        ($data) = $_ =~ m/(\d+)%/;
82      }
83      elsif ($_ =~ m/Index/) {
84        ($index) = $_ =~ m/(\d+)%/;
85      }
86      elsif ($_ =~ m/not a NDB node/) { # Error code should take care of this. Check nevertheless
           .
87        print "Unable to get info for node ${node}. Does it exist?";
88        exit(3); # Exit with nagios UNKNOWN code
89      }
90    }
91    # Set warning or critical if thresholds are exceeded
92    if($index > $crit_thresh || $data > $crit_thresh) {
93      $exit_code = 2;
94    }
95    elsif(($index > $warn_thresh || $data > $warn_thresh) && $exit_code == 0) {
96      $exit_code = 1;
97    }
98
99    return "Node_${node}_Data=$data% Node_${node}_Index=$index% ";
100 }
101
102 sub usage {
103   print "check_ndb_mem usage:\n";
104   print "\t--host \t\tThe ndb manager to connect to\n";
105   print "\t--nodes \tThe IDs of the nodes to check, comma separated\n";
106   print "\t--warning \tThe percent of usage threshold for warning. Defaults to 70 if not
          specified\n";
107   print "\t--critical \tThe percent of usage threshold for critical. Defaults to 80 if not
          specified\n";
108
109   return 0;
110 }
111
112 main();
```

H.4 Brukeroppretting for SQL-motorer

script/oracledb.sql

```
--Oracle DB Monitoring user
--http://labs.consol.de/lang/en/nagios/check_oracle_health/

CREATE USER nagios IDENTIFIED BY oradbmon;
GRANT CREATE SESSION TO nagios;
GRANT SELECT any dictionary TO nagios;
GRANT SELECT ON V_$SYSSTAT TO nagios;
GRANT SELECT ON V_$INSTANCE TO nagios;
GRANT SELECT ON V_$LOG TO nagios;
GRANT SELECT ON SYS.DBA_DATA_FILES TO nagios;
GRANT SELECT ON SYS.DBA_FREE_SPACE TO nagios;
```

script/mssqldb.sql

```
--MSSQL DB Monitoring user
--http://labs.consol.de/lang/en/nagios/check_mssql_health/

declare @dbname varchar(255)
declare @check_mssql_health_USER varchar(255)
declare @check_mssql_health_PASS varchar(255)
declare @check_mssql_health_ROLE varchar(255)
declare @source varchar(255)
declare @options varchar(255)
declare @backslash int

/*****************************************************************/
SET @check_mssql_health_USER = '"[Servername|Domainname]\Username"'
SET @check_mssql_health_PASS = 'Password'
SET @check_mssql_health_ROLE = 'Rolename'
/*****************************************************************

PLEASE CHANGE THE ABOVE VALUES ACCORDING TO YOUR REQUIREMENTS

- Example for Windows authentication:
  SET @check_mssql_health_USER = '"[Servername|Domainname]\Username"'
  SET @check_mssql_health_ROLE = 'Rolename'

- Example for SQL Server authentication:
  SET @check_mssql_health_USER = 'Username'
  SET @check_mssql_health_PASS = 'Password'
  SET @check_mssql_health_ROLE = 'Rolename'

!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
It is strongly recommended to use Windows authentication. Otherwise
you will get no reliable results for database usage.
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

********** NO NEED TO CHANGE ANYTHING BELOW THIS LINE **********/

SET @options = 'DEFAULT_DATABASE=MASTER, DEFAULT_LANGUAGE=English'
SET @backslash = (SELECT CHARINDEX('\', @check_mssql_health_USER))
IF @backslash > 0
  BEGIN
    SET @source = ' FROM WINDOWS'
    SET @options = ' WITH ' + @options
  END
ELSE
  BEGIN
    SET @source = ''
    SET @options = ' WITH PASSWORD=''' + @check_mssql_health_PASS + ''',' + @options
  END

PRINT 'create Nagios plugin user ' + @check_mssql_health_USER
EXEC ('CREATE LOGIN ' + @check_mssql_health_USER + @source + @options)
EXEC ('USE MASTER GRANT VIEW SERVER STATE TO ' + @check_mssql_health_USER)
PRINT 'User ' + @check_mssql_health_USER + ' created.'
PRINT ''

declare dblist cursor for
```

```
  select name from sysdatabases WHERE name NOT IN ('master', 'tempdb', 'msdb') open dblist
    fetch next from dblist into @dbname
    while @@fetch_status = 0 begin
      EXEC ('USE [' + @dbname + '] print ''Grant permissions in the db '' + ''"'' + DB_NAME() + ''"'')
      EXEC ('USE [' + @dbname + '] CREATE ROLE ' + @check_mssql_health_ROLE)
      EXEC ('USE [' + @dbname + '] GRANT EXECUTE TO ' + @check_mssql_health_ROLE)
      EXEC ('USE [' + @dbname + '] GRANT VIEW DATABASE STATE TO ' + @check_mssql_health_ROLE)
      EXEC ('USE [' + @dbname + '] GRANT VIEW DEFINITION TO ' + @check_mssql_health_ROLE)
      EXEC ('USE [' + @dbname + '] CREATE USER ' + @check_mssql_health_USER + ' FOR LOGIN ' + @check_mssql_health_USER)
      EXEC ('USE [' + @dbname + '] EXEC sp_addrolemember ' + @check_mssql_health_ROLE + ' , ' + @check_mssql_health_USER)
      EXEC ('USE [' + @dbname + '] print ''Permissions in the db '' + ''"'' + DB_NAME() + ''" granted.''')
      fetch next from dblist into @dbname
    end
close dblist
deallocate dblist
```

H.5 Init-script

script/metricinga_init.d

```
#! /bin/sh
### BEGIN INIT INFO
# Provides:          metricinga
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: metricinga init script
# Description:       An init script for Graphite's processing script metricinga
### END INIT INFO

# Adapted for metricinga by MonKey 2013
# Based on carbon-cache init script by Jeremy Chalmer
#
# Enable with update-rc.d metricinga defaults


# Source init-functions:
. /lib/lsb/init-functions

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
HOST=localhost
OPTIONS="-H localhost"
# Path
METRICINGA_HOME=/opt/graphite/bin

# Name of executable daemon
NAME=metricinga
DESC=metricinga

# Path to Executable
DAEMON=$METRICINGA_HOME/metricinga.py

PIDFILE=/var/run/metricinga.pid

SCRIPTNAME=/etc/init.d/$NAME

# Exit if the package is not installed
if [ ! -x "$DAEMON" ]; then {
    echo "Couldn't find $DAEMON or not executable"
    exit 99
}
fi

# Load the VERBOSE setting and other rcS variables
[ -f /etc/default/rcS ] && . /etc/default/rcS
```

```
47
48 #
49 # Function that starts the daemon/service
50 #
51 do_start()
52 {
53     # Return
54     #   0 if daemon has been started
55     #   1 if daemon was already running
56     #   2 if daemon could not be started
57
58     # Test to see if the daemon is already running - return 1 if it is.
59     start-stop-daemon --start --pidfile $PIDFILE \
60        --test  --exec $DAEMON -- $OPTIONS > /dev/null || return 1
61
62     # Start the daemon for real, return 2 if failed
63     start-stop-daemon --start --pidfile $PIDFILE \
64         --exec $DAEMON -- $OPTIONS > /dev/null || return 2
65 }
66
67 #
68 # Function that stops the daemon/service
69 #
70 do_stop() {
71     # Return
72     #   0 if daemon has been stopped
73     #   1 if daemon was already stopped
74     #   2 if daemon could not be stopped
75     #   other if a failure occurred
76     log_daemon_msg "Stopping $DESC" "$NAME"
77     start-stop-daemon --stop --signal 2 --retry 5 --quiet --pidfile $PIDFILE
78     RETVAL="$?"
79     [ "$RETVAL" = 2 ] && return 2
80
81     # Delete the exisitng PID file
82     if [ -e "$PIDFILE" ]; then {
83         rm $PIDFILE
84     }
85     fi
86
87     return "$RETVAL"
88 }
89
90
91 # Display / Parse Init Options
92 case "$1" in
93   start)
94    [ "$VERBOSE" != no ] && log_daemon_msg "Starting $DESC" "$NAME"
95     do_start
96     case "$?" in
97       0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
98       2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
99     esac
100  ;;
101  stop)
102   [ "$VERBOSE" != no ] && log_daemon_msg "Stopping $DESC" "$NAME"
103    do_stop
104    case "$?" in
105      0|1) [ "$VERBOSE" != no ] && log_end_msg 0 ;;
106      2) [ "$VERBOSE" != no ] && log_end_msg 1 ;;
107    esac
108  ;;
109  restart)
110    log_daemon_msg "Restarting $DESC" "$NAME"
111    do_stop
112      case "$?" in
113        0|1)
114        do_start
115        case "$?" in
116          0) log_end_msg 0 ;;
117          1) log_end_msg 1 ;; # Old process is still running
118          *) log_end_msg 1 ;; # Failed to start
119        esac
120     ;;
121    *)
```

```
      # Failed to stop
    log_end_msg 1
    ;;
  esac
  ;;
  status)
      if [ -s $PIDFILE ]; then
          pid=`cat $PIDFILE`
          kill -0 $pid >/dev/null 2>&1
          if [ "$?" = "0" ]; then
              echo "$NAME is running: pid $pid."
              RETVAL=0
          else
              echo "Couldn't find pid $pid for $NAME."
              RETVAL=1
          fi
      else
          echo "$NAME is stopped (no pid file)."
          RETVAL=1
      fi
  ;;
  *)
  echo "Usage: $SCRIPTNAME {start|stop|restart|status}" >&2
  exit 3
  ;;
esac
:
```

H.6 Statusvindu

script/index.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/
    xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Icinga Status</title>

  <link href="style.css" rel="stylesheet">
  <link href="//netdna.bootstrapcdn.com/twitter-bootstrap/2.2.1/css/bootstrap-combined.min.css"
       rel="stylesheet">

  <script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script> <!-- TODO:
       Download to local -->
  <script src="//netdna.bootstrapcdn.com/twitter-bootstrap/2.2.1/js/bootstrap.min.js"></script>
  <script src="js/external.js"></script>
   <script src="Highcharts/js/highcharts.js"></script>
  <script src="js/clock.js"></script>
  <script src="js/jquery.marquee.js"></script>

</head>

<body>
  <div id="contents">
    <div id="top">
      <div id="serverroom_climate">
        <div id="temperature">
          <div id="data"></div>
        </div>

        <div id="humidity">
          <div id="data"></div>
        </div>
      </div> <!-- end serverroom_climate -->

      <div id="clock">
          <div id="Date"></div>
        <ul>
          <li id="hours"></li>
          <li id="point">:</li>
```

```
          <li id="min"></li>
          <li id="point">:</li>
          <li id="sec"></li>
        </ul>
      </div>

      <div id="footprints">
        <div id="graph"></div>
      </div>
      <div id="feed"><div class="marquee" id="data" scrollamount="0"></marquee></div></div>
    </div> <!-- end top -->

    <div id="nagioscontainer"></div>
  </div> <!-- end contents -->
</body>
```

script/nagdash.php

```
<?php
require_once 'config.php';
error_reporting(!E_NOTICE);

// For authenticating agains Icinga Classic
$context = stream_context_create(array(
    'http' => array(
    'header'  => "Authorization: Basic " . base64_encode("$classic_username:$classic_password")
    )
));

// Definitions for nagios states
$nagios_host_status = array(0 => "UP", 1 => "DOWN", 2 => "UNREACHABLE");
$nagios_service_status = array(0 => "OK", 1 => "WARNING", 2 => "CRITICAL", 3 => "UNKNOWN");

// Definitions for nagios states to css-classes
$nagios_host_status_colour = array(0 => "status_green", 1 => "status_red", 2 => "status_yellow");
$nagios_service_status_colour = array(0 => "status_green", 1 => "status_yellow", 2 => "status_red", 3 => "status_grey");

// Host-query part for url used in retreiving data from icinga-web API, all hosts which are down or unreachable
$hostQuery = "filter[OR(HOST_CURRENT_STATE|=|1;HOST_CURRENT_STATE|=|2)]/
          columns[HOST_ID|HOST_CURRENT_CHECK_ATTEMPT|HOST_OUTPUT|HOST_NAME|
              HOST_LAST_STATE_CHANGE|HOST_CURRENT_STATE|HOST_PROBLEM_HAS_BEEN_ACKNOWLEDGED|
               HOST_SCHEDULED_DOWNTIME_DEPTH|HOST_NOTIFICATIONS_ENABLED|HOST_MAX_CHECK_ATTEMPTS]/
          order(HOST_CURRENT_STATE;DESC)/
          countColumn=HOST_ID/
          authkey=$APIkey/json";

// Service-query, all services in critical or warning state and host is UP (0)
$serviceQuery = "filter[AND(HOST_CURRENT_STATE|=|0;OR(SERVICE_CURRENT_STATE|=|1;SERVICE_CURRENT_STATE|=|2))]/
            columns[SERVICE_ID|HOST_SCHEDULED_DOWNTIME_DEPTH|SERVICE_OUTPUT|
                SERVICE_NOTIFICATIONS_ENABLED|SERVICE_SCHEDULED_DOWNTIME_DEPTH|
                 SERVICE_PROBLEM_HAS_BEEN_ACKNOWLEDGED|SERVICE_NAME|HOST_NAME|
                      SERVICE_CURRENT_STATE|HOST_NAME|HOST_CURRENT_STATE|
                      SERVICE_LAST_STATE_CHANGE|
                 SERVICE_MAX_CHECK_ATTEMPTS|SERVICE_CURRENT_CHECK_ATTEMPT]/
            order(SERVICE_CURRENT_STATE;DESC)/
            countColumn=SERVICE_ID/
            authkey=$APIkey/json";

// Totals queries
$hostTotalQuery = "filter/columns[HOST_CURRENT_STATE]/countColumn=HOST_ID/authkey=$APIkey/json";
$serviceTotalQuery = "filter/columns[SERVICE_CURRENT_STATE]/countColumn=SERVICE_ID/authkey=$APIkey/json";

$hostPriority = array();
$servicePriority = array ();

$hostArray=array();
$serviceArray=array();
```

```
46
47 // Totals for hosts and services in each state
48 $host_summary = array();
49 $service_summary = array();
50
51 $known_hosts = array();
52 $known_services = array();
53 // Holds services and hosts in warning or critical state, that are not acknowledged
54 $down_hosts = array();
55 $broken_services = array();
56
57 // Run query against Icinga web API on specified APIhost
58 function fetchStatus($APIhost,$target, $query) {
59   $query = preg_replace('/(\s)+/', '', trim($query)); // Remove whitespace
60   $statusDecoded = json_decode(file_get_contents("http://$APIhost/icinga-web/web/api/{$target}/
         $query"));
61    if(!$statusDecoded) {
62     die("<div class='status_red'>Error with $target query </div>");
63   }
64   if($statusDecoded->success !== 'true')
65     die("<div class='status_red'>{$statusDecoded->errors[0]}</div>");
66
67   $vars = get_object_vars($statusDecoded);
68
69   return($vars);
70 }
71
72 $hosts = fetchStatus($APIhost, "host", $hostTotalQuery);
73 $services = fetchStatus($APIhost, "service", $serviceTotalQuery);
74
75 $hostArray = fetchStatus($APIhost, "host", $hostQuery);
76 $serviceArray = fetchStatus($APIhost, "service", $serviceQuery);
77
78 foreach($hosts['result'] as $host) {
79   $host_summary[$host->{'HOST_CURRENT_STATE'}] += 1;
80 }
81
82 foreach($services['result'] as $service) {
83   $service_summary[$service->{'SERVICE_CURRENT_STATE'}] += 1;
84 }
85
86 // Used in sorting of hosts, based on state and priority variable
87 function hostCompare($x, $y) {
88   //Unreachable is placed after Critical (1 supersedes 2)
89   if($x['state'] != 1 && $y['state'] == 1) return 1;
90
91   //If both is down, place the one with highest priority first
92   if($x['state'] == 1 && $y['state'] == 1)
93     return ($x['priority'] < $y['priority']) ? 1 : -1;
94 }
95
96 //  Used in sorting of services, based on state and priority variable
97 function serviceCompare($x, $y) {
98   // Critical (2) before Warning (1)
99   if ($x['state'] < $y['state']) return 1;
100
101   // If both is Critical, place the one with highest priorty first
102   if($x['state'] == 2 && $y['state'] == 2)
103     return ($x['priority'] < $y['priority']) ? 1 : -1;
104 }
105
106 // Loop and check if host is ack'd or not, then populate given array
107
108 foreach($hostArray["result"] as $host_details) {
109   $host_attributes = get_object_vars($host_details);
110   if (($host_attributes['HOST_PROBLEM_HAS_BEEN_ACKNOWLEDGED'] > 0) || ($host_attributes['
         HOST_SCHEDULED_DOWNTIME_DEPTH'] > 0) || ($host_attributes['HOST_NOTIFICATIONS_ENABLED']
         == 0) ) {
111     $array_name = "known_hosts";
112   } else {
113     $array_name = "down_hosts";
114   }
115    $hostPriorityQuery = 'filter[AND(OR(HOST_ID|=|' . $host_attributes['HOST_ID'] . ');
         HOST_CUSTOMVARIABLE_NAME|=|PRIORITY)]/columns[HOST_ID|HOST_NAME|
         HOST_CUSTOMVARIABLE_VALUE]/order(HOST_CURRENT_STATE;DESC)/countColumn=HOST_ID/authkey='
```

```
        . $APIkey . '/json';

  $priorityResult = fetchStatus($APIhost, 'host', $hostPriorityQuery);
  $priority = get_object_vars($priorityResult['result'][0]);
  // Populate either known_hosts or down_hosts array with key => value
  array_push($$array_name, array(
    "priority" => (isset($priority)) ? $priority['HOST_CUSTOMVARIABLE_VALUE'] : 0,
    "hostname" => $host_attributes['HOST_NAME'],
    "state" => $host_attributes['HOST_CURRENT_STATE'],
    "duration" => $host_attributes['HOST_LAST_STATE_CHANGE'],
    "detail" => $host_attributes['HOST_OUTPUT'],
    "current_attempt" => $host_attributes['HOST_CURRENT_CHECK_ATTEMPT'],
    "max_attempts" => $host_attributes['HOST_MAX_CHECK_ATTEMPTS'],
    "is_hard" => ($host_attributes['HOST_CURRENT_CHECK_ATTEMPTS'] >= $host_attributes['
        HOST_MAX_CHECK_ATTEMPTS']) ? true : false,
    "is_downtime" => ($host_attributes['HOST_SCHEDULED_DOWNTIME_DEPTH'] > 0) ? true : false,
    "is_ack" => ($host_attributes['HOST_PROBLEM_HAS_BEEN_ACKNOWLEDGED'] > 0) ? true : false,
    "is_enabled" => ($host_attributes['HOST_NOTIFICATIONS_ENABLED'] > 0) ? true : false,
  ));
}

// Loop and check if service is ack'd or not, then populate given array
foreach($serviceArray['result'] as $service_detail) {
  $service_attributes = get_object_vars($service_detail);
  $servicePriorityQuery = 'filter[AND(OR(SERVICE_ID|=|' . $service_attributes['SERVICE_ID'] .
                  ');SERVICE_CUSTOMVARIABLE_NAME|=|PRIORITY)]/
                  columns[SERVICE_ID|SERVICE_CUSTOMVARIABLE_VALUE]/order(SERVICE_CURRENT_STATE;
                      DESC)/authkey=' . $APIkey . '/json';
  $priorityResult = fetchStatus($APIhost, 'service', $servicePriorityQuery);
  $priority = get_object_vars($priorityResult['result'][0]);

  //if the host is OK, AND the service is NOT OK.
  // Sort the service into the correct array. It's either a known issue or not.
  if ( ($service_attributes['SERVICE_PROBLEM_HAS_BEEN_ACKNOWLEDGED'] > 0) || (
      $service_attributes['SERVICE_SCHEDULED_DOWNTIME_DEPTH'] > 0) || ( $service_attributes['
      SERVICE_NOTIFICATIONS_ENABLED'] == 0 ) || ($service_attributes['
      HOST_SCHEDULED_DOWNTIME_DEPTH'] > 0) ) {
     $array_name = "known_services";
  } else {
     $array_name = "broken_services";
  }

  // Populate either known_services or broken_hosts array with key => value
  array_push($$array_name, array(
    "priority" => (isset($priority)) ? $priority['SERVICE_CUSTOMVARIABLE_VALUE'] : 0,
    "hostname" => $service_attributes['HOST_NAME'],
     "service_name" => $service_attributes['SERVICE_NAME'],
    "state" => $service_attributes['SERVICE_CURRENT_STATE'],
    "duration" => $service_attributes['SERVICE_LAST_STATE_CHANGE'],
     "detail" => $service_attributes['SERVICE_OUTPUT'],
    "current_attempt" => $service_attributes['SERVICE_CURRENT_CHECK_ATTEMPT'],
    "max_attempts" => $service_attributes['SERVICE_MAX_CHECK_ATTEMPTS'],
    "is_hard" => ($service_attributes['SERVICE_CURRENT_CHECK_ATTEMPT'] >= $service_attributes['
        SERVICE_MAX_CHECK_ATTEMPTS']) ? true : false,
    "is_downtime" => ($service_attributes['SERVICE_SCHEDULED_DOWNTIME_DEPTH|'] > 0) ? true :
        false,
    "is_ack" => ($service_attributes['SERVICE_PROBLEM_HAS_BEEN_ACKNOWLEDGED'] > 0) ? true :
        false,
    "is_enabled" => ($service_attributes['SERVICE_NOTIFICATIONS_ENABLED'] > 0) ? true : false,
  ));
}
?>

<div id="host_section">
  <div id="frame">
    <div class="section">
      <p class="totals">
        <span class="section_title">Hosts</span>
        <span class="total_count">
        <?php foreach($host_summary as $state => $count) {
          echo "<span class='{$nagios_host_status_colour[$state]}'>{$count}</span> "; } ?>
        </span>
      </p>
  <?php
      if (count($down_hosts) > 0):
```

```
182         uasort($down_hosts, 'hostCompare');
183       ?>
184       <table id="broken_hosts" class="widetable">
185       <tr><th>Hostname</th><!--<th>Pri</th>--><th width="150px">State</th><th>Since</th><!--<th
              >Attempt</th>--><th>Detail</th></tr>
186   <?php foreach($down_hosts as $host) {
187         echo "<tr id='host_row' class='{$nagios_host_status_colour[$host['state']]}'>";
188         echo "<td>{$host['hostname']}</td>";
189         //  echo "<td>{$host['priority']}</td>";
190         echo "<td>{$nagios_host_status[$host['state']]}</td>";
191         echo "<td>{$host['duration']}</td>";
192         //echo "<td>{$host['current_attempt']}/{$host['max_attempts']}</td>";
193         echo "<td class=\"desc\">{$host['detail']}</td>";
194         echo "</tr>";
195       }
196       ?>
197       </table>
198     <?php
199       else:
200     ?>
201       <table class='widetable status_green'><tr><td><b>All hosts OK</b></td></tr></table>
202     <?php
203       endif;
204
205     if (count($known_hosts) > 0): ?>
206       <div class='known_problems' id='known_host_probems'>
207         <p class='totals'><span class='known_section_title'>Known Host Problems</span></p>
208           <table class='widetable known_hosts'>
209           <th>Hostname</th><th>State</th><th>Since</th><th>Detail</th></tr>
210       <?php foreach($known_hosts as $this_host) {
211             $status_text = false;
212             if ($this_host['is_ack']) $status_text = "ack";
213             if ($this_host['is_downtime']) $status_text = "downtime";
214             if (!$this_host['is_enabled']) $status_text = "disabled";
215             echo "<tr id='host_row' class='{$nagios_host_status_colour[$this_host['state']]}'>"
                  ;
216             echo "<td>{$this_host['hostname']}</td>";
217             echo "<td>{$nagios_host_status[$this_host['state']]} " . ($status_text ? "(
                  $status_text)" : "") . "</td>";
218             echo "<td>{$this_host['duration']}</td>";
219             echo "<td class=\"desc\">{$this_host['detail']}</td>";
220             echo "</tr>";
221           }
222      ?>
223         </table>
224       </div><!-- known_problems end-->
225   <?php endif; ?>
226     </div> <!-- section end -->
227   </div> <!-- frame end -->
228 </div> <!-- host_section end -->
229
230 <div id="service_section">
231   <div id="frame">
232     <div class="section">
233       <p class="totals">
234         <span class="section_title">Services</span>
235         <span class="total_count">
236         <?php foreach($service_summary as $state => $count) {
237           echo "<span class='{$nagios_service_status_colour[$state]}'>{$count}</span> ";
238         } ?>
239         </span>
240       </p>
241
242   <?php if (count($broken_services) > 0): uasort($broken_services, 'serviceCompare');
243   ?>
244       <table class="widetable" id="broken_services">
245       <tr><th width="20%">Hostname</th><!--<th>Pri</th>--><th width="50%">Service</th><th width
              ="15%">State</th><th width="15%">Since</th><!--<th width="5%">Attempt</th>--></tr>
246   <?php foreach($broken_services as $service) {
247         if ($service['is_hard']) { $soft_tag = ""; } else { $soft_tag = "(soft)"; }
248           echo "<tr class='{$nagios_service_status_colour[$service['state']]}'>";
249           echo "<td>{$service['hostname']}</td>";
250 //          echo "<td>{$service['priority']}</td>";
251           echo "<td>{$service['service_name']} - {$service['detail']}</td>";
252           echo "<td>{$nagios_service_status[$service['state']]} {$soft_tag}</td>";
```

```
            echo "<td>{$service['duration']}</td>";
            //echo "<td>{$service['current_attempt']}/{$service['max_attempts']}</td>";
            echo "</tr>";
        }
  ?>
      </table>
  <?php else: ?>
      <table class="widetable status_green"><tr><td><b>All services OK</b></td></tr></table>
  <?php endif;
      if (count($known_services) > 0): ?>
      <div class="known_problems" id="known_service_problems">
        <p class="totals"><span class="known_section_title">Known Service Problems</span></p>
        <table class="widetable" id="known_services">
        <tr><th width="30%">Hostname</th><th width="37%">Service</th><th width="18%">State</th
            ><th width="10%">Since</th><!--<th width="5%">Attempt</th>--></tr>
  <?php   foreach($known_services as $service) {
            if ($service['is_ack']) $status_text = "ack";
               if ($service['is_downtime']) $status_text = "downtime";
               if (!$service['is_enabled']) $status_text = "disabled";
            echo "<tr class='{$nagios_service_status_colour[$service['state']]}'>";
               echo "<td>{$service['hostname']}</td>";
                 echo "<td>{$service['service_name']}</td>";
              echo "<td>{$nagios_service_status[$service['state']]} ({$status_text})</td>";
               echo "<td>{$service['duration']}</td>";
             //echo "<td>{$service['current_attempt']}/{$service['max_attempts']}</td>";
             echo "</tr>";
          }
   ?>

        </table>
      </div>
  <?php endif; ?>
    </div>
  </div>
</div>
```

script/netbotz.php

```
<?php
/*
 * Contacts Icinga web API and serves data from services
 * Check Netbotz Humid and Check Netbotz Temp as json data
 * Author Monkey, 2013
 */

require_once 'config.php';

// Queries Icinga API and return array of sensor values of set type
function check_sensor($query, $type, $dummydata) {
  // Serve dummy data if we are in test
  if($dummydata) {
    return array(array('name' => 1, 'data' => 19), array('name' => 2, 'data' => 18), array('
        name' => 4, 'data' => 23), array('name' => 5, 'data' => 24), array('name' => 3, 'data'
        => 24));
  }
  // Query the API
  $result = json_decode(file_get_contents($query));
  $result_obj = $result->result[0];
  // Get value for each sensor
  // Data looks like: 'Sensor_MM:1_Humidity'=16%;;;99;0;_'Sensor_MM:2_Humidity'=15%;;;99;0;_'
      Sensor_MM:5_Humidity'=15%;;;99;0;_'Sensor_MM:4_Humidity'=15%;;;99;0;_
  preg_match_all('/Sensor_MM:(\d+)_' . $type . '\'=(\d+)|{%};/', $result_obj->SERVICE_PERFDATA,
      $matches);
  $result = array();

  for($i = 0; $i < count($matches[1]); $i++) {
    $sensor = array();
    $sensor['name'] = $matches[1][$i];
    $sensor['data'] = $matches[2][$i];

    array_push($result, $sensor);
  }
  return $result;
}
```

```
33
34 if(isset($_GET['type'])) {
35   $query = $type = null;
36   if($_GET['type'] == 'Temperature') {
37       $query = 'http://' . $APIhost . '/icinga-web/web/api/service/filter[(SERVICE_NAME%7C=%7
           CCheck%20Netbotz%20Temp)]/countColumn=SERVICE_ID/authkey=' . $APIkey . '/json';
38       echo json_encode(check_sensor($query, 'Temperature', $dummydata), JSON_NUMERIC_CHECK);
39   }
40   elseif ($_GET['type'] == 'Humidity') {
41     $query = 'http://' . $APIhost . '/icinga-web/web/api/service/filter[(SERVICE_NAME%7C=%7
         CCheck%20Netbotz%20Humid)]/countColumn=SERVICE_ID/authkey=' . $APIkey . '/json';
42     echo json_encode(check_sensor($query, 'Humidity', $dummydata), JSON_NUMERIC_CHECK);
43   }
44 }
```

script/rss.php

```
1  <?php
2  /*
3   * Downloads an rss-feed and serve the item titles as
4   * a json array
5   */
6  require_once('config.php');
7  if(isset($_GET['url'])) {
8    $data = array();
9    $xml = simplexml_load_string(file_get_contents($_GET['url']));
10
11   if(!$xml)
12     die('unable to load rss feed');
13
14   foreach($xml->channel->item as $item) {
15     // Hardcoded in the rss-feed for driftsmeldinger, we don't want to include it
16       if($item->title != 'Det er for tiden ingen driftsmeldinger.') {
17         array_push($data, (string) $item->title);
18       }
19   }
20   if($dummydata) {
21     echo json_encode(array("Det er planlagt driftsarbeid pØ mail-servere natt til 03. mai
           mellom 02.00 og 02:30", "Det kan oppleves noe treghet pØ terminalservere. Det jobbes
           med Ø utbedre dette."));
22   } else {
23     echo json_encode($data);
24   }
25 } else {
26   echo "Must be called with url as get param";
27 }
```

script/image.php

```
1  <?php
2  /**
3   * Servers an illustration of the server room based on the number of sensors in a column.
4   * Author: MonKey, 2013
5   * Last edited: 24.04.2013
6   */
7
8  // Exit if we have no get param
9  if(!(isset($_GET['sensor_cols']) && is_numeric($_GET['sensor_cols'])))
10   die('Needs to be called with a numeric get param of sensor_cols');
11
12 $base    = 'img/room.png'; // The start image
13 $add     = 'img/room_n.png'; // For each column over 1, we paste in one of this
14 $no_add   = $_GET['sensor_cols'] - 1; // number of columns to add
15
16 // Create images
17 $base_image = imagecreatefrompng($base);
18 $add_image  = imagecreatefrompng($add);
19
20 // Get original dimensions
21 list($base_width, $base_height) = getimagesize($base);
22 list($add_width, $add_height) = getimagesize($add);
23
24 // Create output image
```

```php
$new = imagecreatetruecolor(($base_width+($add_width*$no_add)), $base_height);
// Set output image transparent
imagealphablending($new, false );
imagesavealpha($new, true);

// Copy our base to the output
imagecopy($new, $base_image, 0, 0, 0, 0, $base_width, $base_height);

// Add columns by copying add_image to output
for($i = 0; $i < $no_add; $i++)
  imagecopy($new, $add_image, $base_width + ($i*$add_width), 0, 0, 0, $add_width, $add_height);

// Serve up the image
header('Content-Type: image/png');
imagepng($new);
```

script/footprints.php

```php
<?php

/**
 * Get statistics from footprints database and serve as json
 *
 * Author: MonKey, 2013
 * Last edited: 24.04.2013
 */

require_once 'db_config.php';
require_once 'config.php';

// Setup connection to database and SELECT Footprints if we are in production
if(!$dummydata) {
  mssql_connect($servername,$dbusername,$dbpassword) or die("Unable to connect to server");
  mssql_select_db("Footprints");
}

// Queries Footprints and returns the result as key => value
function fetchAmount($dummydata, $query, $type) {
  if($dummydata) {
    return array($type => rand(0,90));
    }

  $result = mssql_query($query);
  $amount = mssql_fetch_assoc($result);

   return array($type => $amount['computed']);
}

if(isset($_GET['type'])) {
  if($_GET['type'] == 'Graph') {
      $resultArray = array();
      //Submitted and closed today
      $closedQuery = "SELECT COUNT(mrID) FROM dbo.MASTER2 WHERE DATEDIFF(day, mrSUBMITDATE,
          GETDATE()) = 0 AND mrSTATUS = 'Closed'";
      //Received today
      $receivedQuery = "SELECT COUNT(mrID) FROM dbo.MASTER2 WHERE DATEDIFF(day, mrSUBMITDATE,
          GETDATE()) = 0";
      //Open overall
      $openQuery = "SELECT COUNT(mrID) FROM dbo.MASTER2 WHERE mrSTATUS != 'Closed' AND mrSTATUS
           != '_SOLVED_' AND mrSTATUS != '_DELETED_'";
      //Submitted overall, closed today
      $closedAllQuery = "SELECT COUNT(mrID) FROM dbo.MASTER2 WHERE DATEDIFF(day, mrUPDATEDATE,
          GETDATE()) = 0 AND  mrSTATUS = 'Closed'";

      //Merge the results from each query for json-dump
      $resultArray = array_merge(fetchAmount($dummydata, $closedQuery, 'Closed'), fetchAmount(
          $dummydata, $receivedQuery, 'Received'),
                      fetchAmount($dummydata, $closedAllQuery, 'ClosedAll'), fetchAmount(
                          $dummydata, $openQuery, 'Open'));

      echo json_encode($resultArray, JSON_NUMERIC_CHECK);
  }
}
```

script/config.php

```php
<?php

$classic_username = 'icingaadmin';  // Icinga classic credentials
$classic_password = 'password';
$APIkey       = 'api_key'; // API-key to Icinga web
$APIhost      = 'localhost';  // host with Icinga web
$dummydata      = false;      // Serve dummy data
```

script/db_config.php

```php
<?php

$dbusername ='SQLuser';
$dbpassword ='SQLpassword';
$servername ='SQLhost';
```

script/external.js

```javascript
// Generic ajax function
function ajax(arg, func, url) {
  var errorID = 'error_' + func.name;
  $.ajax({
    type: 'GET',
    url: url,
    data: arg
  }).done(function(data, text, jqXHR) {
    if($('#' + errorID).length > 0)
      $('#' + errorID).remove();
    func(data);
  }).fail(function(jqXHR, textStatus, errorThrown) {
    var error = '<div id="' + errorID + '" class="status_red">Error contacting: ' + url + ' for function: ' + func.name + '</div>';
    if($('#' + errorID).length == 0)
      $("#top").append(error);
  });
}

// Callback function for temperature
// Sets all temperature sensor values
function temp(sensor_data) {
  var front_layout = [2, 1];  // The netbotz sensor IDs for the front row
  var back_layout = [5, 4]; // and the back
  var columns = Math.max(front_layout.length, back_layout.length); // Largest row is used for drawing columns

  var front_data = new Array();
  var back_data = new Array();

  // Loop all sensors and add the value to the array where it belongs (taken from the layouts)
  $.each($.parseJSON(sensor_data), function(index, sensor) {
    if(front_layout.indexOf(sensor.name) > -1) {
      front_data[front_layout.indexOf(sensor.name)] = sensor.data;
      }
      else if(back_layout.indexOf(sensor.name) > -1) {
      back_data[back_layout.indexOf(sensor.name)] = sensor.data;
      }
  });

  // Generate HTML
  var html = "<div id=\"back\">";
  for(var i in back_data) {
    html += "<span id=\"temp_sensor_back_" + i + "\" class=\"temp_sensor_back\">" + back_data[i] + "&degC</span>";
  }

  html += "</div><div id=\"front\">";

  for(var i in front_data) {
    html += "<span id=\"temp_sensor_front_" + i + "\" class=\"temp_sensor_front\">" + front_data[i] + "&degC</span>";
  }
```

```
50
51    html += "</div>";
52
53    var room = new Image(); // Set up background image for the div
54    room.onload = function() {
55      $('#serverroom_climate').css('width', this.width);
56      $('#serverroom_climate').css('height', this.height);
57      $('#serverroom_climate').css('background-image', 'url(' + this.src + ')');
58    };
59
60    // Set the background image to be one with the right number of columns
61    room.src = "image.php?sensor_cols=" + columns;
62    // Set the sensor values in the data div
63    $('#serverroom_climate #temperature #data').html(html);
64  }
65
66  // Callback function for humidity
67  // Sets the mean humidity value
68  function humid(sensor_data) {
69    var total = 0;
70    var sensors = $.parseJSON(sensor_data);
71
72    // Generate the total by looping all sensors
73    $.each(sensors, function(index, sensor) {
74      total += sensor.data;
75      });
76    var mean = Math.round(total/sensors.length);
77
78    $('#serverroom_climate #humidity #data').html(mean + '%');
79  }
80
81  function getGraphStats(graphStats) {
82  //  var barArray = new Array();
83    var stats = $.parseJSON(graphStats);
84
85    $('#graph').highcharts({
86      chart: {
87        renderTo: 'graph',
88        width: 400,
89        height: 175,
90        animation: false,
91        defaultSeriesType: 'column',
92        marginTop: 26
93      },
94      title: {
95        text: null
96      },
97      xAxis: {
98        categories: ['Closed(today)', 'Received(today)', 'Closed(overall)', 'Active(overall)']
99      },
100     yAxis: {
101       title: {
102         text: null
103       },
104       endOnTick: false,
105       max: Math.max(stats.Open, stats.Closed, stats.ClosedAll, stats.Received)
106     },
107     legend: {
108       enabled: false
109     },
110     labels: {
111       rotation: -45,
112       align: 'right'
113     },
114     series: [{
115       name: 'Amount',
116       data: [{ y: stats.Closed,
117              color: '#32CD32'},
118            { y: stats.Received,
119         color: '#FF0000'},
120           { y: stats.ClosedAll,
121         color: '#32CD32'},
122           { y: stats.Open,
123         color: '#FF8F00'}
124          ]
```

```
      }],
    plotOptions: {
      column: {
        dataLabels: {
          enabled: true,
          style: {
            fontSize: "16px",
            lineHeight: "auto"
          }
        },
        animation: false,
        pointWidth: 30,
          borderWidth: 1
      }
    },

    tooltip: {
      enabled: false
    },
    credits: {
      enabled: false
    }
  });
}

// Callback function for rss feed
// Sets the #feed #data div.
// If all feed items are longer than the div we make the text scroll
function feed(data) {
  var html = '';

  $.each($.parseJSON(data), function(key, val) {
    html += '&#x25cf; ' + val + '    ';
  });

  $("#feed #data").html(html);
  var feedWidth = $("#feed #data")[0].scrollWidth;
  var areaWidth = $("#feed #data").width();

  if(feedWidth > areaWidth) {
    $("#feed #data").css("overflow", "hidden");
    $("#feed #data").marquee({
      speed: 30000
    });
  }
}

function nagdash(data) {
  $('#nagioscontainer').html(data);
}

// Set up all our functions to run and rerun
function load() {
  (function contDelay() {
    ajax(null, nagdash, 'nagdash.php');
    ajax({type: "Temperature"}, temp, 'netbotz.php');
    ajax({type: "Humidity"}, humid, 'netbotz.php');
    ajax({type: "Graph"}, getGraphStats, 'footprints.php');
    setTimeout(contDelay, 200000);
  })();

  (function feedDelay(){
    ajax({url: 'https://portal.hedmark.org/driftsfeed.php'}, feed, 'rss.php');
    setTimeout(feedDelay, 100000);
  })();
}

$(document).ready(function() {
  $.ajaxSetup({ cache: false });
  load();
});
```

script/style.css

```
h3                  { margin-top: 3px; margin-bottom: 3px; font-size: 1.5em }
body                { font-family: "HelveticaNeue-Medium", Helvetica, Arial, sans-serif; margin: 5px; margin-top: 0px; min-width: 900px; }
table,td            { border: none; padding: 2px; border-spacing: 2px; font-size: 1.0em }
table,tr      { text-align: center; }
table table { border:0px;}
table               { border: 1px solid #c6c6c6; background-color: #F0F0F0; border-collapse: separate;
  *border-collapse: collapse; -webkit-border-radius: 4px;
  -moz-border-radius: 4px; border-radius: 4px; }
  th                  { border: 1px black solid; background-color: #D8D8D8 }
  .widetable          { width: 99%;}
  .bold               { font-weight: bold; }
  .status_green       { background-color: #269926; color: white; padding: 3px }
  .status_red         { background-color: #FF4040; color: white; padding: 3px }
  .status_yellow      { background-color: #FFDE40; color: black; padding: 3px }
  .status_grey        { background-color: #444444; color: white; padding: 3px }
  .status_blue    { background-color: #50BBD7; color: white; padding: 3px }
  .desc               { font-size: 0.8em }
  .left               { float: left}
  .totals             { right: 10px; padding: 5px 0px; border: 1px #848484 solid; background: #F0F0F0; width: 99%;
                -webkit-border-radius: 4px; -moz-border-radius: 4px; border-radius: 4px;
                margin-top: 5px; margin-bottom: 5px; text-align: right; }

.section_title { float: left; font-size: 1.2em; font-weight: bold; margin-left: 1%; }
.known_section_title { text-align: left; display: block; font-size: 1.2em; font-weight: bold; margin-left: 1%; }
.total_count { margin-right: 1%; }
.known_problems { opacity:0.3; font-size: 1.0em;}

#contents { margin: 15px }
#host_section { float: left; width: 50%; margin-left: -5px; }
#service_section { float: right; width: 50%; margin-right: -5px; }

/* Top section */

/* Date and clock section */
#clock { float: right; text-align: center;}
#clock ul { margin: 0 auto; list-style:none; }
#clock ul li { display: inline; font-size: 2em; }
#clock #Date { padding-bottom: 10px; }

/* Server room graphics and info */
#serverroom_climate { position: absolute; background url('img/room.png') no-repeat; }
#temperature #data #back {  position: absolute; margin-left: 20px; font-size: 1.4em; margin-top: 18px;}
#temperature #data #front { position: absolute; margin-left: 20px; font-size: 1.4em; margin-top: 100px; }
#temperature #data #back span, #temperature #data #front span { padding-right: 20px; }

#humidity { background: url('img/humid.png') center center no-repeat; height: 100%;}
#humidity #data { font-size: 1.4em; position: relative; text-align: center; height: 100%; top: 60px; }

#footprints { margin: 0 auto; width: 400px; }

#feed { white-space: nowrap;  width: 90%; margin: 0 auto; }
#feed #data { overflow: auto;  font-size: 1.2em;}
```

H.7 Differanse på endrede plug-ins

vedlegg/diff.pl

```
ï»¿
# diff check_asa_vpn.pl orig/check_asa_sessions.pl
<                 'sessions' => '1.3.6.1.4.1.9.9.392.1.3.1.0',
<                 'maxSessions' => '1.3.6.1.4.1.9.9.392.1.1.1.0'
---
>                 'sessions' => '.1.3.6.1.4.1.9.9.147.1.2.2.2.1.5.40.6',
109c107
<         my ($sessions, $maxSessions);
---
>         my ($sessions);
120c118
<         my $result = $session->get_request ($snmp_string{sessions}, $snmp_string{maxSessions
    });
---
>         my $result = $session->get_request ($snmp_string{sessions});
124c122
<         unless (defined ($status{sessions}) && defined ($status{maxSessions})) {
---
>         unless (defined ($status{sessions})) {
129c127
<         return ($status{sessions}, $status{maxSessions});
---
>         return ($status{sessions});
137c135
< # Get the status for sessions
---
> # Get the status for the active and standby unit
139,140c137
< my ($sessions, $maxSessions) = &status_request ($hostname, $community, $debug, $timeout,
    $retries);
< my $percentUse = ceil(($sessions / $maxSessions)*100);
---
> my ($sessions) = &status_request ($hostname, $community, $debug, $timeout, $retries);
142c139
< # Determine status to icinga
---
> # Determine status of cluster
144,145c141,142
< if ($percentUse < $warnning) {
<         print "OK - Cisco ASA VPN sessions: $sessions of $maxSessions used ($percentUse%) |
    vpn_users=$sessions\n";
---
> if ($sessions < $warnning) {
>         print "OK - Cisco ASA sessions:$sessions\n";
147,148c144,145
< } elsif ($percentUse > $warnning && $percentUse < $critical) {
<         print "Warning - Cisco ASA VPN sessions: $sessions of $maxSessions used ($percentUse
    %) | vpn_users=$sessions\n";
---
> } elsif ($sessions > $warnning && $sessions < $critical) {
>         print "Warning -Cisco ASA sessions:$sessions\n";
151c148
<         print "Critical - Cisco ASA VPN sessions: $sessions of $maxSessions used ($percentUse
    %) | vpn_users=$sessions\n";
---
>         print "Critical - Cisco ASA sessions:$sessions\n";
154c151
<         print "Unknown - Cisco ASA VPN $sessions | vpn_users=$sessions\n";
---
>         print "Unknown - Cisco ASA $sessions\n";
156a154,155



# diff check_iftraffic64.pl orig/check_iftraffic64.pl
88d87
< my $no_percent;       #added 20130502 by MonKey
119c118
< my $TRAFFIC_FILE = "/var/cache/icinga/traffic/";
```

```
---
> my $TRAFFIC_FILE = "/usr/local/nagios/libexec/traffic/";
169,170d167
<         #added 20130502 by MonKey
<         "n|nopercent" => \$no_percent,
350,357c347,350
<                         if($iface_speed == 0) {         # default to 100mbps
<                                 $iface_speed = 100 * 1000 * 1000;
<                         } else {
<                                 # number returned but NOT greater than zero - bad output
<                                 # try 32 bit speed counter
<                                 $iface_speed = $response->{ $snmpIfSpeed32 . "." .
    $iface_number };
<                                 debugout ("INTERFACE using 32 bit speed counters:
    $iface_speed", "2");
<                         }
---
>                         # number returned but NOT greater than zero - bad output
>                         # try 32 bit speed counter
>                         $iface_speed = $response->{ $snmpIfSpeed32 . "." . $iface_number };
>                         debugout ("INTERFACE using 32 bit speed counters: $iface_speed", "2")
    ;
449a443
>
469,480d462
< # Added by MonKey, 2013
< # Lets us use absolute numbers (in mbps) instead of percentages
< my $in_test;
< my $out_test;
< if($no_percent) {
<         $in_test = $in_ave / (1000*1000);
<         $out_test = $out_ave / (1000*1000);
< } else {
<         $in_test = $in_ave_pct;
<         $out_test = $out_ave_pct;
< }
<
505d486
<
507c488
< if ( ( $in_test > $crit_usage ) or ( $out_test > $crit_usage ) or ( $if_status != 1 ) ) {
---
> if ( ( $in_ave_pct > $crit_usage ) or ( $out_ave_pct > $crit_usage ) or ( $if_status != 1 ) )
    {
511c492,493
< if (   (( $in_test > $warn_usage ) or ( $out_test > $warn_usage )) && $state eq "OK" )
---
> if (   ( $in_ave_pct > $warn_usage )
>         or ( $out_ave_pct > $warn_usage ) && $state eq "OK" )
625c607
<                         print "Interface $ifdescr = $snmpkey ";  #debug
---
>                         print "$ifdescr = $key / $snmpkey \n";  #debug
747c729
<     Usage: check_iftraffic64.pl -H host [ -C community_string ] [ -i if_index|if_descr ] [ -r
    ] [ -b if_max_speed_in | -I if_max_speed_in ] [ -O if_max_speed_out ] [ -n ] [ -u ] [ -B ]
    [ -A IP Address ] [ -L ] [ -M ] [ -w warn ] [ -c crit ]
---
>     Usage: check_iftraffic64.pl -H host [ -C community_string ] [ -i if_index|if_descr ] [ -r
    ] [ -b if_max_speed_in | -I if_max_speed_in ] [ -O if_max_speed_out ] [ -u ] [ -B ] [ -A
    IP Address ] [ -L ] [ -M ] [ -w warn ] [ -c crit ]
784,785d765
<         -n --nopercent FLAG
<         Sets the thresholds to be in mbps instead of percent



# diff check_netbotz.py orig/check_netbotz.py
<     parser.add_option("-t", "--type", dest="type", default="temp", type="string", help="Test
    Type. Valid values are 'temp' for temprature test, and 'humid' for humidity tests. [Default
    :temp]")
---
>     parser.add_option("-t", "--type", dest="type", default="temp", type="string", help="Test
    Type. Valid values are 'temp' for tempratue test, and 'humid' for humidity tests. [Default:
```

```
    temp]")
65,69c65
<     parser.add_option("--warning_low", dest="warn_l", type="int", help="The low warning
    threshold")
<     parser.add_option("--critical_low", dest="crit_l", type="int", help="The low critical
    threshold")
<     parser.add_option("--warning_high", dest="warn_h", type="int", help="The high warning
    threshold")
<     parser.add_option("--critical_high", dest="crit_h", type="int", help="The high critical
    threshold")
<
---
>
72a69
>
76,80c73
<
<     if options.warn_l == None or options.crit_l == None or options.warn_h == None or options.
    crit_h == None:
<                 print "Error: high and low warning and critical thresholds must be supplied."
<                 sys.exit(RET_CODES["UNKNOWN"])
<
---
>
105c98
<         test_name = "Temperature"
---
>         test_name = "Tempratue"
160a154
>
162d155
<         sensor_val = int(return_values[key]["value"])
164,172c157,158
<         if sensor_val < options.crit_l:
<             ret_code = RET_CODES["CRITICAL"]
<             desc += " - The %s value is LOWER than allowed treshold. " % test_name
<         elif sensor_val > options.crit_h:
<             ret_code = RET_CODES["CRITICAL"]
<             desc += " - The %s value is HIGHER than allowed treshold. " % test_name
<         elif sensor_val < options.warn_l:
<             if ret_code == RET_CODES["OK"]:
<                 ret_code = RET_CODES["WARNING"]
---
>         if return_values[key]["value"] < return_values[key]["low_value"]:
>             ret_code = on_error_retcode
174,179c160,164
<         elif sensor_val > options.warn_h:
<             if ret_code == RET_CODES["OK"]:
<                ret_code = RET_CODES["WARNING"]
<             desc += " - The %s value is HIGHER than allowed treshold. " % test_name
<         else:
<             desc += " - The %s value is OK. " % test_name
---
>         elif return_values[key]["value"] > return_values[key]["high_value"]:
>             ret_code = on_error_retcode
>             desc += " - The %s value is HIGHER than allowed treshold. " % test_name
>
>         desc += " - The %s value is OK. " % test_name
185,186d169
<     elif ret_code == RET_CODES["WARNING"]:
<         desc = "WARNING: "+desc
188c171
<         desc = "CRITICAL: "+desc
---
>         desc = "ERROR: "+desc
192,193c175
<         sensor_name = return_values[key]["name"].replace(" ", "_")
<         desc += "'%s_%s'=%s%s;;;%s;%s; " % (sensor_name,
---
>         desc += "'%s %s'=%s%s;;;%s;%s; " % (return_values[key]["name"],
198a181
>


# diff check_ndbd.pl orig/check_ndbd.pl
```

```
199
200
201 < my $check_command = '/usr/sbin/ndb_mgm';
202 ---
203 > my $command_prefix = '/opt/mysql/bin/';
204 > my $check_command = 'ndb_mgm';
205 40d40
206 < my $host=undef;
207 44,45c44
208 < GetOptions(                "check_command=s"                => \$check_command,
209 <                           "host=s"                => \$host,
210 ---
211 > GetOptions(        "check_command=s"                => \$check_command,
212 88c87
213 <                 print "OK - Acceptable number of nodes connected. | mgm_nodes=$mgm_value
    ndb_nodes=$ndb_value sql_nodes=$sql_value\n";
214 ---
215 >                 print "OK - Acceptable number of nodes connected.\n";
216 104c103
217 <         open(NS, $check_command. " " . $host ." -e $check_expression |") || die "Command
    failed! Check the script cause it sucks...\n";
218 ---
219 >         open(NS, $command_prefix.$check_command." -e $check_expression |") || die "Command
    failed! Check the script cause it sucks...\n";
220 138c137
221 <         open(NS, $check_command . " " . $host ." -e $check_expression |") || die "Command
    failed!  Is your path (prefix) correct?\n";
222 ---
223 >         open(NS, $command_prefix.$check_command." -e $check_expression |") || die "Command
    failed!  Is your path (prefix) correct?\n";
224
225
226
227
228 < $np->add_arg(
229 <   spec => 'interval|i=s',
230 <   help => "-i, --interval=<seconds>\n"
231 <     . '   Interval in seconds between 1st and last xml sample',
232 <   required => 0,
233 < );
234 <
235 229d220
236 < my $interval = $np->opts->interval;
237 233c224
238 < $debug_timeshift = (!$interval) ? 10 : $interval;
239 ---
240 > $debug_timeshift = 10;
241 341,342d331
242 <
243 <
244 346c335
245 <         my $perf;
246 ---
247 >     my $perf = {};
248 348,354c337
249 <         if ($command eq 'CPU' or 'MEM' and defined($subcommand)) {
250 <         if (uc($subcommand) eq "USAGE") {
251 <                         return $rrd;
252 <                 }
253 <         }
254 <
255 <         # get newest perf data
256 ---
257 >     # get newest perf data
258 356,357c339,340
259 <                 if ($time < $row->{timestamp}) {
260 <                 $time = $row->{timestamp};
261 ---
262 >         if ($time < $row->{timestamp}) {
263 >             $time = $row->{timestamp};
264 361,362c344
265 <         return $perf;
266 <
267 ---
268 >     return $perf;
```

```
373,374c355,356
<         my $timestamp = parse_datetime_iso8601($host->get_servertime()) - $debug_timeshift;
<         return get_latest_perfdata($host, $timestamp);
---
>     my $timestamp = parse_datetime_iso8601($host->get_servertime()) - $debug_timeshift;
>     return get_latest_perfdata($host, $timestamp);
386c368
<             my $timestamp = parse_datetime_iso8601($host->get_servertime()) - $debug_timeshift
    ;
---
>     my $timestamp = parse_datetime_iso8601($host->get_servertime()) - $debug_timeshift;
486c468,469
<         my $usage = 0;
---
>     my $perf = $uuid ? get_latest_host_by_uuid_perfdata($xen, $hostname) :
    get_latest_host_by_name_perfdata($xen, $hostname);
>     my $usage = 0;
491c474
<         my $perf = $uuid ? get_latest_host_by_uuid_perfdata($xen, $hostname) :
    get_latest_host_by_name_perfdata($xen, $hostname);
---
>
494,499c477,482
<                         foreach my $row (@{$perf}) {
<                           $usage += $row->{data}->{cpu_avg}->{$rolluptype};
<                           $i++;
<                         }
<                         $usage = $usage / $i;
<                         $usage = simplify_number($usage * 100);
---
>             # get all cpu values with keys: cpu0, cpu1, ..., cpu7, ...
>             while (my $val = $perf->{"cpu$i"}) {
>                 $usage += $val->{$rolluptype};
>                 $i++;
>             }
>             $usage = simplify_number($usage / $i * 100) if ($i > 0);
535c518
<     my $usage = 0;
---
>     my $usage = 'nan';
542d524
<         my $i = 0;
543a526
>
546,558c529,532
<                         my $allocatedMem = 0;
<                         foreach my $row (@{$perf}) {
<               $allocatedMem = $row->{data}->{memory_total_kib}->{$rolluptype};
<                           my $tempUsage = $allocatedMem - $row->{data}->{memory_free_kib}->{
    $rolluptype};
<                           $usage += $tempUsage;
<                           $i++;
<             }
<                         $usage = $usage / $i;
<                         my $percentUsage = int($usage / $allocatedMem * 100);
<             my $mbusage = simplify_number($usage / 1024);
<             $output = "mem: usage = " . $mbusage . " MB (" . $percentUsage . "%)";
<             $np->add_perfdata(label => "used", value => perfvalue($mbusage), uom => 'MB',
    threshold => $np->threshold);
<             $res = $np->check_threshold(check => $percentUsage);
---
>             $usage = simplify_number($perf->{memory_total_kib}->{$rolluptype} / 1024) if (
    exists($perf->{memory_total_kib}));
>             $output = "mem: usage = " . $usage . " MB";
>             $np->add_perfdata(label => "used", value => perfvalue($usage), uom => 'MB',
    threshold => $np->threshold);
>             $res = $np->check_threshold(check => $usage);
624c598
<                 $send += $value->{tx} / 1024 if (exists($value->{tx}));
---
>         $send += $value->{tx} / 1024 if (exists($value->{tx}));
758c732
<                         $localtime_diff = time() - parse_datetime_iso8601($host->
    get_server_localtime());
---
```

```
336 >         $localtime_diff = time() - parse_datetime_iso8601($host->get_server_localtime());
337
338
339
340
341 # diff metricinga.py orig/metricinga.py
342 < import MySQLdb
343 <     PERFDATA TIL DATABASE
344 <     if len(check_result['SERVICEPERFDATA'])!= 0:
345 <         conn = MySQLdb.connect (host = "localhost",
346 <                                 user = "root",
347 <                                 passwd = "<Password>",
348 <                                 db = "perfdata")
349 <
350 <         cursor = conn.cursor ()
351 <
352 <         cursor.execute( 'INSERT INTO servicedata(TIMET,HOSTNAME,SERVICEDESC, SERVICEPERFDATA)
      values("%s","%s","%s","%s")' % (check_result['TIMET'],check_result['HOSTNAME'],
     check_result['SERVICEDESC'],check_result['SERVICEPERFDATA']) )
```

Vedlegg I

E-post

E-poster av relevans for rapporten. Svar er fjernet av hensyn til mottaker.

Fra:bundy.icinga@gmail.com
Emne:Retrieving a specific custom variable through The Icinga-Web REST API
Dato:Fri, 3 May 2013 14:03:25 +0200
Til:icinga-users@lists.sourceforge.net

Hey!

I am wondering if it is possible to retrieve the column
HOST_CUSTOMVARIABLE_VALUE or SERVICE_CUSTOMVARIABLE_VALUE based on a given
custom variables name.

I have a custom variable named _PRIORITY and want to get the value of this
variable, and not the other custom variables the object has in the result
of the query.

So something like
...columns[HOST_CUSTOMVARIABLE_VALUE|where|HOST_CUSTOMVARIABLE_NAME|=|PRIORITY|...]

Fra:oyvind@sigerstad.com
Emne:Nagdash Licence
Dato:Thu, 4 Apr 2013 11:39:24 +0200
Til:laurie@denness.net

Hi!

We're doing a bachelor thesis where we implement Icinga for the local
county. We are currently looking at solutions for how to implement
dashboard functionality to give a quick overview to help desk workers.
We looked at Nagdash and it seems like a good starting point for what we
want to do, so we were wondering what license, if any, it is published
under.

Sincerely,
Øyvind Sigerstad

Fra:MonKey
Emne:Mail to HWGroup
Dato:Februar
Til:Mail to HWGroup

Hello! We are looking to implement temperature and humidity in a "Nagios"
Environment for our bachelors project, can you give us some prices and
info for equipment you would recommend?

We have looked at this one: HWg-STE: Ethernet thermometer,
With this we also need probes.
Can you ship to Norway?

Regards
-Monkey

Fra:"Nils Slåen" <nils.slaen@hig.no>
Emne:Weathergoose spørsmål
Dato:tor, januar 31, 2013, 11:17
Til:mvold@uninett.no

Hei!

Vi er en bachelorgruppe på 3 som skal overvåke infrastruktur med
Nagios/Icinga.

Så er vi kommet til steget der vi skal kjøpe inn sensor for luftfuktighet
og temperatur. Vi ser at du har sist endret denne:
https://openwiki.uninett.no/gigacampus:weathergoose

Så vi lurte på hvor du kjøpte IT Watchdog produkter, og evt hvordan
erfaringene med disse har vært?

Mvh Nils Slåen
Høgskolen i Gjøvik

Fra:"Nils Slåen" <nils.slaen@hig.no>
Emne:Oracle Health Question
Dato:fre, april 12, 2013, 15:50
Til:gerhard.lausser@consol.de

Hello from Norway!

We are a bachelor group from Norway doing our project on Icinga, by now we
have about 200 hosts and 600 service checks, and using your MySQL, MSSQL,
and your Oracle plugins for a couple of database servers.

Now my question is, we have some different Oracle servers, they are
running a different number of Oracle instances, Should we monitor every
instance? Which means checks like cache hit and connection time should be
checked on every "Instance" Not only each server?

In that case, is there a way to check all instances for one server? Don't
really want to script alot of services, or use check multi.

Regards, Nils Slåen
Monkey bachelor group.
Gjøvik University College

Fra:"Nils Slåen" <nils.slaen@hig.no>
Emne:Server-rom overvåkning
Dato:ons, februar 6, 2013, 16:59
Til:svein-inge.kvalo@hedmark.org

Hei.

Produktene vi har funnet som ser mest lovende ut er ITwatchdogs, det er
også disse som er benyttet av Uninett. De er rackmonterbar, støtter snmp
v3 og kan utvides med flere sensorer i ettertid. Prisforskjellen går på
hvilke sensorer som er innebygd i selve enheten og hvor mange porter den
har for sensorer. Sensorene er basert på 1-wire, som gjør disse
forholdsvis billige.
Alle produktene har 1 års garanti.

1. http://www.itwatchdogs.com/product-detail-minigoose_xp_ii-10.html
Innebygd temperatursensor. 16 digitale innganger og 3 analoge.
Digitale sensorer er f.eks temp, luftfuktighet, luftflyt etc, mens analoge
er f.eks vann, røyk, dørposisjon og strøm.

2. http://www.itwatchdogs.com/product-detail-weathergoose_ii-1.html
Innebygd temperatur, luftfuktighet, luftflyt, lys og lyd. 5 digitale
innganger og 3 analoge. Kan utvides med portsplittere.

3. http://www.itwatchdogs.com/product-detail-watchdog_100-67.html
Denne har kun en digital innganger, og kan ikke splittes i flere, her er
det en innebygget temperaturføler, og en luftfuktighetsmåler.

Ekstra temperatur sensor
http://www.itwatchdogs.com/product-detail-temperature_sensor-4.html

Alle disse vil fungere ypperlig til oppgaven. Hvilken som velges er
avhengig av hvilke sensorer dere ønsker å ha i boksen, og utvidbarheten
ved tanke på flere sensorer i ettertid.

-Nils

Fra:bundy.icinga@gmail.com
Emne:Question about check_xenapi.pl plugin for Nagios
Dato:Sat, 11 May 2013 17:22:42 +0200
Til:op5-users@lists.op5.com

Hello

I have started using the check_xenapi.pl plugin and I am wondering about the results it is returning. From what I have understood it uses the data with the latest timestamp from this code block:

```
sub get_latest_perfdata {
    my ($obj, $timestamp) =3D @_;
    my $rrd =3D $obj->get_rrd_update($timestamp);
    my $perf =3D {};
    my $time =3D 0;
    # get newest perf data
    foreach my $row (@{$rrd}) {
        if ($time < $row->{timestamp}) {
            $time =3D $row->{timestamp};
            $perf =3D $row->{data};
        }
    }
    return $perf;
```

The plugin also per default asks the RRD-database for data in the following interval (quoted from the documentation on RRDs on XenServer):

*Every 5 seconds for the duration of 10 minutes

And for this interval it says the following:

*The sampling that takes place every 5 seconds records actual data points

So my question is really if this is intended behavior, because I have not found any options to specify which interval I want data from, or if this should be averaged. The returned results will be actual datapoints, which can hard to set tresholds on since a spike can happen in that 5 second interval.

I might be totally off here, and misunderstood the code / or use of the plugin.

Thanks in advance.

Kind regards,
Bjoern-Erik