jamk.fi

Incident response -työkalujen vertailu

Petteri Viitanen

Opinnäytetyö
Joulukuu 2017
Tekniikan ja liikenteen ala
Insinööri (AMK), tietoverkkotekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Viitanen, Petteri	Opinnäytetyö, AMK	Joulukuu 2017
	Sivumäärä 63	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: Kyllä

Työn nimi
Incident response -työkalujen vertailu

Tutkinto-ohjelma
Tietoverkkotekniikan koulutusohjelma

Työn ohjaaja(t)
Tero Kokkonen, Mika Rantonen

Toimeksiantaja(t)
JYVSECTEC

Toimeksiantajan tavoitteena oli selvittää kahden avoimen lähdekoodin ohjelmiston soveltuvuutta jokapäiväiseen Incident Response -toimintaan. Toimeksiantaja määritti ohjelmistoiksi Cyphon ja YETI.

Vertailu tehtiin ohjelmistojen asennuksen ja käyttöönoton, ominaisuuksien sekä käytettävyyden perusteella. Työssä vertailtiin myös ohjelmistoja tiiminhallinnan perusteella sekä sen mukaan, että kuinka hyvin näistä ohjelmistoista saadaan tietoja ulos kolmansiin järjestelmiin API-rajapintojen avulla.

YETI osoittautui yksinkertaisemmaksi, tehokkaammaksi ja käyttöönotoltaan nopeammaksi vaihtoehdoksi, mutta toiminnallisuudeltaan rajatuksi. Cyphon sisälsi enemmän toiminallisuutta kuten front end GUI:n, mutta Cyphonin tehokas käyttö vaati paljon enemmän opettelua. Myös asennus- ja käynnistämisprosessi olivat hitaampia.

Cyphonista ei löytynyt toiminnallisuutta manuaalisten ilmoitusten lisäämiseen. Sen vuoksi ohjelmistosta kirjoitettiin kattava raportti, joka käsittelee automaattista tiedonkeräämistä ja automaattisten hälytysten luomista.

YETI:llä ajettiin toimeksiantajan kanssa suunniteltu esimerkkitapaus, jossa listattiin erilaisia poikkeustapahtumia ja toimijoita ohjelmistoon. YETI suoriutui tehtävästä hyvin, mutta ohjelmiston toiminnallisuus oli hyvin rajallista. Kaikkia toimeksiantajan hakemia toiminnallisuuksia ei löytynyt ohjelmistosta.

Työn lopputuloksena saatiin vertailtua sekä manuaalista, että automaattista toimintaperiaatetta, sekä niiden soveltuvuutta Incident Response -toimintaan. Nämä kaksi toiminnallisuutta eroavat suuresti toisistaan, ja molemmille on omat käyttötarkoituksensa.

Avainsanat (asiasanat)
Incident Response, Incident Management, API, Cyphon, YETI, Poikkeamienhallinta

Muut tiedot (salassa pidettävät liitteet)

jamk.fi

Description

Author(s)	Type of publication	Date
Viitanen, Petteri	Bachelor's thesis	December 2017
		Language of publication: Finnish
	Number of pages: 63	Permission for web publication: Yes

Title of publication

Comparison between Incident Response programs

Degree programme

Information Technology

Supervisor(s)

Kokkonen, Tero; Rantonen, Mika

Assigned by

JYVSECTEC

Abstract

The goal of the assignment was to determine how two open source software programs fit for everyday Incident Response work. The assigned software programs were Cyphon and YETI.

The comparison was carried out in terms of installation, initialization, usability and functionality. The further goals were to compare these programs regarding team management and determine their API functionality for third party software integration.

YETI proved to be simple, efficient and the option that initialized faster. However, its functionality was limited. Cyphon had more built-in functionality such as the front-end GUI. However, in order to use Cyphon efficiently, further studying is required. The installation and initialization phases of Cyphon were also much slower.

Cyphon was found not to have the functionality to manually add incidents. For that reason, a comprehensive report on the automated functionality of Cyphon was written instead.

A test-case was designed with the assigner, which comprised of manual addition of incidents and different units to YETI software. YETI completed the task, however its functionality was very limited. The software did not have all the required functionality the assigner was seeking.

Both manual- and automated functionality were put to the test in the assignment. Each functionality has their cases of use and they were not directly comparable.

Keywords/tags (subjects)

Incident Response, Incident Management, API, Cyphon, YETI

Miscellaneous (Confidential information)

Sisältö

Kuviot

Taulukot

Lyhenteet

API	Application Programming Interface
BCP	Business Continuity Plan
BIA	Business Impact Analysis
CIA	Confidentiality, Integrity, Availability
CIRT	Computer Incident Response Team
CSIRT	Computer Security Incident Response Team
DRP	Disaster Recovery Plan
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
IR	Incident Response
IRP	Incident Response Plan
JSON	Javascript Object Notation
MITM	Man in the middle
NIST	National Institute of Standards and Technology
OODA	Observe, Orient, Decide, Act
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
SLA	Service Level Agreement
SQL	Structured Query Language
TTP	Tactics, Techniques and Procedures
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
XML	Extensible Markup Language

1 Johdanto

JYVSECTEC tarjoaa tutkimus-, kehitys-, koulutus- ja harjoituspalveluita yrityksille ja organisaatioille. JYVSECTEC ylläpitää ja kehittää RGCE-kybertoimintaympäristöä. (Tietoa meistä n.d.)

Toimeksiantajan tavoitteena oli selvittää kahden eri open source -tuotteen soveltuvuutta Incident Response -toimintaan. Incident Response tarkoittaa suomennettuna poikkeamiin vastaamista ja Incident Management poikkeamienhallintaa. Ohjelmiksi valittiin Cyphon ja YETI. Cyphon on Dunbar-yrityksen kehittämä ilmainen ohjelmisto. YETI on monien eri käyttäjien kehittämä avoin ohjelmisto.

Teoriaosuudessa määritetään teoreettinen pohja poikkeamienhallinnalle ja tilannetietoisuudelle sekä myös taustatiedot käytettävissä oleville teknologioille.

Riskinhallinnan ohjelmia vertaillaan niiden toiminnallisuuden, käytettävyyden, asennettavuuden sekä sen mukaan, että kuinka hyvin tietoa voidaan viedä kolmansiin osapuoliin API-rajapinnan kautta. Vertailtaessa API-rajapintaa, toimeksiantajan pääpainona oli GET-metodilla saatavien tietojen hakeminen järjestelmästä. Tätä kautta tietoa voidaan hakea, lukea ja se voidaan visualisoida kolmannen osapuolen apusovelluksessa.

Lopullinen vertailu tehtiin luomalla kuvitteellinen esimerkkitapaus, jossa yrityksen verkkoon on päästy hyökkäämään. Vertailussa keskitytään ohjelmistojen monipuolisuuteen tiiminhallinnan osalta sekä arvioidaan kuinka hyvin ohjelmistot osaavat muodostaa ns. keissin. Tapauksen selvittäminen alkaa aina puutteellisen informaation pohjalta. Uusia tekijöitä ja merkkejä kompromisseista löydetään tutkinnan edetessä. Tällöin on hyvä, että esimerkkitapausta eli keissiä voidaan täydentää. Keissille pitäisi olla ns. aikajanatoiminto, jolloin tapauksen eri vaiheet saadaan linkitettyä yhteen kronologisessa järjestyksessä.

2 Teknologiat ja taustatutkimus

Työssä verrattiin kahta eri poikkeamienhallinnan ohjelmistoa. Käytettävissä olevat teknologiat vaikuttavat ohjelmistojen asennettavuuteen, käyttöönoton helppouteen sekä ominaisuuksien laajuuteen.

2.1 Tutkimusmenetelmät

Kvantitatiivinen tutkimus on faktapohjaista, objektiivista, numeroihin perustuvaa, tuloskeskeistä ja siinä tehdään monia eri kokeita, jotta saadaan varmistettua testaustuloksen tarkkuus. (Räsänen n.d. 4.)

Kvalitatiivinen tutkimus perustuu kokonaiskuvaan, subjektiivisiin mielipiteisiin ja tulkinnan varaisuuteen. Työssä ei ole mitattavia elementtejä ja lopputulos on subjektiivinen, joten siksi työssä käytetään kvalitatiivista tutkimusta. (Räsänen n.d. 4.)

2.2 API

2.2.1 Yleistä

API tulee sanoista ”Application Programming Interface”. Suomennettuna se tarkoittaa ohjelmointirajapintaa. Ohjelmointirajapinnalla yritys voi avata osan palveluaan ohjelmistokehittäjien käyttöön. Näin ohjelmistokehittäjät pystyvät kommunikoimaan rajallisesti yrityksen tarjoaman palvelun kanssa. Tiedonsaanti on kuitenkin rajallista ja ennalta määritettyä, eli API:n käyttöönotto ei avaa koko palvelua ja sen tietoja ulkomaailmaan. Samaan tapaan kuin tiedonsiirtoprotokolla määrittää tiedonvaihdon standardit tietoliikenteessä, API määrittää tiedonvaihdon standardit kahden sovelluksen välillä. Tiedonvaihtoa varten API määrittää säännöt sille mitä protokollaa käytetään, mitkä ovat input- ja output-muotoilusäännöt tiedolle, sekä sen missä muodossa data tuodaan ulos sovelluksesta. (Brajesh 2017, kpl 1 Overview.)

Historiallisesti API on tarkoittanut ohjelmointirajapintoja käyttöjärjestelmissä sekä ohjelmointikirjastojen tarjoamia ohjelmointirajapintoja. Kolmas ja nykyään yleisin API:n muoto on web API. Tehokkaiden älypuhelimien ja tablettien yleistyttyä tarve

rakentaa lisää API-kirjastoja sovelluskehitystä varten kasvoi. Nykyään web API:t on rakennettu käyttäen REST-teknologioita. (Brajesh 2017, kpl 1 The Evolution of API's.)

Toiminnallisuuden kannalta on kriittistä, ettei ohjelmointirajapintaan, datan tulo- ja vientimuotoon, eikä tuodun datan käsittelyformaattiin tehdä muutoksia. Muuten sovellukset voivat lakata toimimasta. Sen vuoksi ohjelmointirajapintaa voidaan pitää teknillisenä sopimuksena kommunikoinnin säännöistä kahden sovelluksen välillä. Tämän vuoksi on myös syytä määrittää SLA-sopimus ohjelmointirajapintaan liittyen. (Brajesh 2017, kpl 1 Defining an API and its characteristics.)

2.2.2 Erilaiset API:t

On olemassa kolmea eri tyyppistä API:a (Brajesh 2017, kpl 1 Types of APIs.)

- Yksityinen API
- Partner API
- Julkinen API

Julkinen API on avoimesti internetissä kaikkien käytettävissä. Esimerkkejä julkisesta API:sta ovat mm. Facebookin, Googlen, Twitterin ja Amazonin tarjoamat API:t. Julkiset API:t tarjoavat tiedonlähteen, joiden tarjoamien toiminnallisuuksien avulla ohjelmistokehittäjät voivat luoda innovatiivisia sovelluksia. Hyvin rakennettu julkinen API voi luoda lisää kohdesovelluksia, tuoda lisää asiakkaita yritykselle sekä nostaa yrityksen markkina-arvoa. Julkiselle API:lle on syytä tarjota dokumentaatio, jossa kuvataan API:n toiminnallisuus. (Brajesh 2017, kpl 1 Types of APIs.)

Yksityiset, tai sisäiset (internal) API:t ovat vain rajatun yhteisön käytössä. Esimerkkinä tästä voisi olla ohjelmoijan luoma rajapinta yrityksen sovellukselle. Partner API:t, eli kumppaneiden väliset API:t ovat tyypillisesti B2B-mallin mukaisesti käytössä kahden yrityksen välillä. Partner API:n avulla voidaan siirtää tietoa kahden yrityksen välillä paljastamatta kuitenkaan alla olevaa ohjelmistokoodia tai back-end palvelin-infrastruktuuria. (Brajesh 2017, kpl 1 Types of APIs.)

2.2.3 Esimerkkejä API toiminnallisuudesta

Taulukossa 1 on kuvattu suurimpien julkisten ohjelmointirajapintojen toiminnallisuuksia.

Taulukko 1. Julkiset API:t ja niiden toiminnallisuus (Brajesh 2017, kpl 1 Examples of popular APIs.)

API	**Toiminnallisuudet**
Twitter	Twitterin REST API:n avulla voidaan lukea tietoa Twitterin käyttäjistä, heidän Twitter-etusivustaan sekä status-päivityksistä. Sen avulla voidaan myös luoda status-päivityksiä. Haku-API:n avulla voidaan hakea tarkemmin status-päivityksiä Twitterin sisältä. Streaming API:n avulla voidaan tarjota jatkuva reaaliaikainen tiedonsyöttö.
Instagram	Instagram API:n avulla voidaan hakea kuvia Instagram-palvelusta ja linkittää niitä omille kotisivuille.
Amazon	Amazonin API:n avulla voidaan mm. tehdä ostoksia ja tarjota mainoksia sovelluksen sisällä. Ostokset tehdään käyttäen Amazonin omaa Amazon Pay -maksujärjestelmää.
Google	Googlen API:n avulla voidaan yhdistää googlen eri palveluihin, kuten haku-, käännös-, kartat- ja gmail-palveluihin. Google API:n avulla voidaan esim. kirjautua sovelluksiin käyttämällä Google-tiliä, eikä tällöin tarvitse luoda erillistä käyttäjätunnusta.
Facebook	Facebookin API:n avulla Facebook-tiliä voidaan käyttää kirjautumiseen ja hakea käyttäjäprofiili-tietoja.
Youtube	Youtube API on osa Googlen API-tarjontaa. Youtube API:n avulla voidaan lisätä videoita ja livestream-tarjontaa verkkosivuille. Analytics-palvelun avulla voidaan hakea videoiden katselukertoihin liittyvää tietoa. Lisäksi palvelu tarjoaa rajapinnan Youtuben käytössä olevaan dataan.

2.3 API-teknologiat

2.3.1 SOAP

SOAP tulee sanoista Simple Object Access Protocol. Sen avulla voidaan välittää järjesteltyä informaatiota jaetussa ympäristössä, joka ei kuitenkaan ole keskitetysti hallittu. SOAP käyttää XML-teknologiaa viestitysmuotonsa. SOAP:n kaksi tärkeintä suunniteltua toiminnallisuutta ovat yksinkertaisuus sekä laajennettavuus. (Gudgin, Hadley, Mendelsohn, Moreau, Nielsen, Karmarkar, Lafon 2007.)

SOAP-viesti kulkee tyypillisesti HTTP:n ylitse, mutta SOAP voi käyttää myös SMTP-protokollaa tai muuta vastaavaa Layer 7-tason viestintäprotokollaa. SOAP-viesti koostuu kirjekuoresta (envelope), jonka sisällä ovat otsikot (headers), viestikenttä (body)

sekä myös virheviesti-kenttä (fault). Viestikenttä sisältää sen varsinaisen informaation jonka haluamme lähettää. Viestikenttä käyttää XML-muotuilua. (Brajesh 2017, kpl 1 The Difference Between a Web Service and a Web API.)

2.3.2 REST

REST tulee sanoista Representational State Transfer. Roy Fielding (2000) määritteli perusteet REST-arkkitehtuurille vuoden 2000 tohtorintyössään.

REST on arkkitehtuuri, jolle ei ole absoluuttisen tarkkoja sääntöjä, mutta noudattaa tiettyjä ohjemalleja ja rajoituksia. REST pohjautuu tilattomaan asiakas-palvelin vuorovaikutusmalliin. Tilaton malli tarkoittaa sitä, että jokaisessa asiakkaan pyynnössä on oltava kaikki tarvittava tieto, jolla serveri osaa käsitellä pyynnön. Tällöin serveri ei siis tallenna mitään tietoja tiedonsiirrosta. (Fielding 2000)

Osana REST-arkkitehtuuria asiakas tallentaa saamansa tiedon säilöön välimuistiin. Tällä mallilla pyritään vähentämään jatkuvaa tiedonsiirtoa. Tällöin tulee olla tiedossa, että onko vastauksessa saatu data säilöttävissä olevaa dataa vai ei. Jos data on säilöttävissä, tällöin asiakaslaite voi käyttää tätä dataa uudelleen samanlaisiin pyyntöihin. Ongelmana voi olla se, jos säilötty data eroaa suuresti asiakaslaitteen ja palvelimen välillä. REST arkkitehtuurille on myös ominaista yhtenäinen tiedonsiirtomalli. Palvelin ja asiakas noudattavat samaa standardoitua tiedonkäsittelyn muotoa. (Fielding 2000)

2.3.3 JSON

JSON-standardi on tekstipohjaisen tiedon esitysmuoto, joka on standardoitu RFC 7159-dokumentissa. JSON-tiedostot käyttävät .json-päätettä.

JSON data-arvot koostuvat kahdesta eri osasta: avaimesta ja arvosta. Avaimen arvon on oltava tekstitietotyyppiä ja avaimen on oltava lainausmerkkien sisällä. JSON tietotyypit ovat teksti (string), luku, objekti, taulukko (array), totuus/väärä tai tyhjä (null). Objektit laitetaan aaltosulkeiden ”{}” sisään. Taulukoiden tieto laitetaan hakasulkeiden ”[]” sisään. Tietueiden arvot erotellaan toisistaan pilkuilla. (JSON Syntax n.d.)

Kuviossa 1 on esimerkki JSON syntaksista.

```
{
        "name":"Petteri",
        "age":24,
        "phones":["Apple","Nokia"],
        "phone": {
                "home":1234,
                "work":5678,
        }
}
```

Kuvio 1 JSON syntaksin esimerkki

JSON sisältää monia hyötyjä verrattuna XML-kieleen. (JSON vs XML n.d.)

- JSON ei käytä lopetus-tagia, eli esim. </html>
- JSON:n esitysmuoto on lyhyempi
- JSON on nopeampi lukea ja kirjoittaa
- JSON voi käyttää taulukkoja tietoarvona
- JSON:n tietoa voidaan parsia Javascript-funktiolla. XML tarvitsee XML-parserin.

2.3.4 XML

XML-kuvauskielen kehitti World Wide Web Consortium eli W3C. XML on standardoitu W3C:n ”Extensible Markup Language (XML) 1.0 (Fifth Edition)”-standardissa. XML-kuvauskielen kehitys alkoi vuonna 1996, eli se on paljon vanhempi teknologia kuin JSON. (Connolly 2003.)

XML noudattaa puurakennetta, joka perustuu lapsi-vanhempi, eli ”parent-child”-rakenteeseen. XML:n rakenne koostuu juuresta, jolla on lapsielementtejä. XML säilöö tiedot elementtien sisällä. Elementin sisältö voi koostua: tekstistä, attribuutista, toisesta elementistä tai edellä mainittujen yhdistelmästä. XML attribuutti viittaa elementille annettuun ominaisuuteen. (XML tree n.d.)

Alla on esimerkki attribuutin käytöstä sekä XML-puun perusrakenteesta.

```
<henkilörekisteri>
        <henkilö sukupuoli=”mies”>Petteri</henkilö>
                <ikä>24</ikä>
                <asuinpaikkakunta>Jyväskylä</asuinpaikkakunta>
        </henkilö>
</henkilörekisteri>
```

Henkilörekisteri on juuri. Sillä on lapsielementti henkilö, joka sisältää attribuutin ”mies”. ”Petteri” on henkilö-elementin tietoarvo, jolle on listattuna 2 lapsielementtiä.

2.4 Web-palvelu

Web-palvelu koostuu resursseista. Resurssien sijainnilla on jokin tietty polku. Resursseihin viitataan joko URN:lla vai URL:lla. URN tulee sanoista Uniform Resource Name ja URL sanoista Uniform Resource Locator. (RFC 2616 1999.)

Absoluuttinen polku viittaa siihen, että URI:ssa annetaan resurssiin viittaava kokonainen polku. Relatiivinen polku olettaa, että kyseinen resurssin lähteen sijainti on tiedossa. Hierarkkinen polku viittaa käytettävissä oleviin aliresursseihin, joiden sijainti kirjoitetaan ”/”-merkkien jälkeen. (RFC 2616 1999.)

Esimerkkinä näistä poluista on UNIX:n tiedostorakenne. /var/www/html on esimerkki absoluuttisesta polusta. ”/”-merkki alussa viittaa UNIX-järjestelmän juuripolkuun. Jos käyttäjä on siirtynyt /var/-kansioon, hän voi tällöin käyttää relatiivista polkua ”www/html” päästäkseen html-kansioon. www- ja html-kansiot ovat /var/-kansion aliresursseja.

Alla on RFC 2616-standardin määrittelemä perusrakenne HTTP-linkille (RFC 2616 1999.):

```
http_URL = "http:" "//" host [ ":" port ] [ abs_path [ "?" query ]]
```

Web-selain olettaa automaattisesti, että resurssia haettaessa käytetään porttia 80. Hyperlinkkiin voidaan sisällyttää eri portti web-sivun osoitteen jälkeen kaksoispisteellä. Kirjoittamalla kysymysmerkin (?) polun jälkeen voidaan linkkiin sisällyttää hakutoiminto. Fragment-toimintoa eli (#)-merkkiä käytetään kun halutaan viitata johonkin resurssissa sijaitsevaan osaan kuten tekstiotsikkoon. (RFC 2616 1999.)

RFC 3986 määrittää hyperlinkin syntaksin auktorisoinnille (RFC 3986 1998.):

```
authority   = [ userinfo "@" ] host [ ":" port ]
```

2.5 HTTP

HTTP on tiedonsiirto-protokolla asiakaslaitteen ja palvelimen välillä. HTTP perustuu pyyntöihin (request) ja vastauksiin (response). HTTP toimii OSI-mallin 7. tasolla eli sovellus-tasolla. HTTP käyttää porttia 80 ja suojattu versio HTTPS porttia 443. (RFC 2616 1999.)

Työn kannalta on tärkeä tutustua siihen, mitä eri HTTP-viestipyyntöjä voidaan tehdä API:in, mitä vastausviestejä saadaan sekä mitä lisätietoja voidaan kuljettaa otsikkokentissä.

2.5.1 HTTP:n metodit

Taulukossa 2 on listattuna tärkeimmät HTTP-viestimetodit. Muita HTTP-viestityypejä ovat TRACE, OPTIONS, HEAD ja CONNECT.

Taulukko 2. HTTP:n pyyntöviestit (RFC 2616 1999.)

Metodi	**Selitys**
GET	GET-metodilla pyydetään hakea URI-linkissä määritettyä resurssia.
POST	POST-metodilla lähetetään pyyntö luoda uusi resurssi serverille, kuten esim. uusi blogi-postaus tai kuvan lähettäminen.
PUT	PUT-metodilla pyydetään serveriltä, että tiettyä olemassa olevaa resurssia muokataan.
DELETE	DELETE-metodilla serveriä pyydetään poistamaan jokin tietty resurssi.

2.5.2 HTTP:n status-viestit

Taulukossa 3 on listattuna tärkeimmät HTTP:n status-viestit.

Taulukko 3. HTTP:n statusviestit (RFC 2616 1999.)

Status	**Selitys**
1xx Informational	Antaa lisätietoa
2xx Succesful	Pyyntö onnistui ja se on vastaanotettu.
200 OK	Pyyntö meni läpi ja siihen on saatu vastaus. Viesti ilmoittaa myös tapauskohtaisesti lisätietoa käytettyyn resurssiin liittyen (GET, POST ym.) Muissa 2xx-viesteissä pyyntö meni myös läpi, mutta pyydettyä sisältöä ei välttämättä saada kokonaisuudessaan.
201 Created	Käyttäjän pyytämä uusi resurssi on luotu.
202 Accepted	Käyttäjän pyyntö on hyväksytty, mutta ei vielä prosessoitu.
3xx Redirection	Uudelleenohjaus. Pyydettyä resurssia varten täytyy suorittaa uudelleenohjaus.
301 Moved Permanently	Viitattu resurssi on siirretty pysyvästi muualle.
302 Found	Viitattu resurssi on siirretty tilapäisesti muualle.
4xx Client Error	Kuvaa käyttäjän tekemää virhettä.
400 Bad Request	Lähetettyä viestipyyntöä ei voitu tunnistaa väärän syntaksin takia.
401 Unauthorized	Ei käyttöoikeutta.
403 Forbidden	Käyttö kielletty.
404 Not found	Resurssia ei löydy tai sen ei haluta olevan julkinen.
405 Method Not Allowed	Pyydetty palvelu on mahdollista toteuttaa, mutta se on poistettu käytöstä
5xx Server Error	Kuvaa virheilmoitusta palvelimen päässä
500 Internal Server Error	Serverin sisäinen virhe
501 Not Implemented	Palvelin ei tue pyydettyä palvelua.
502 Bad Gateway	Serveri toimi välityspalvelimena tai uudelleenohjaajana, mutta ei saanut vastausta seuraavana vuorossa olleelta serveriltä.
503 Service Unavailable	Palvelin ei tilapäisesti pysty käsittelemään pyyntöä tällä hetkellä.
504 Gateway Timeout	Serveri toimi välityspalvelimena tai uudelleenohjaajana, mutta ei saanut vastausta tarpeeksi nopeasti seuraavana vuorossa olleelta serveriltä.

2.5.3 HTTP:n otsikko-kentät

HTTP-viestit sisältävät myös otsikoita. Mahdollisia otsikkotyyppejä ovat

- general-header
- request-header
- response-header

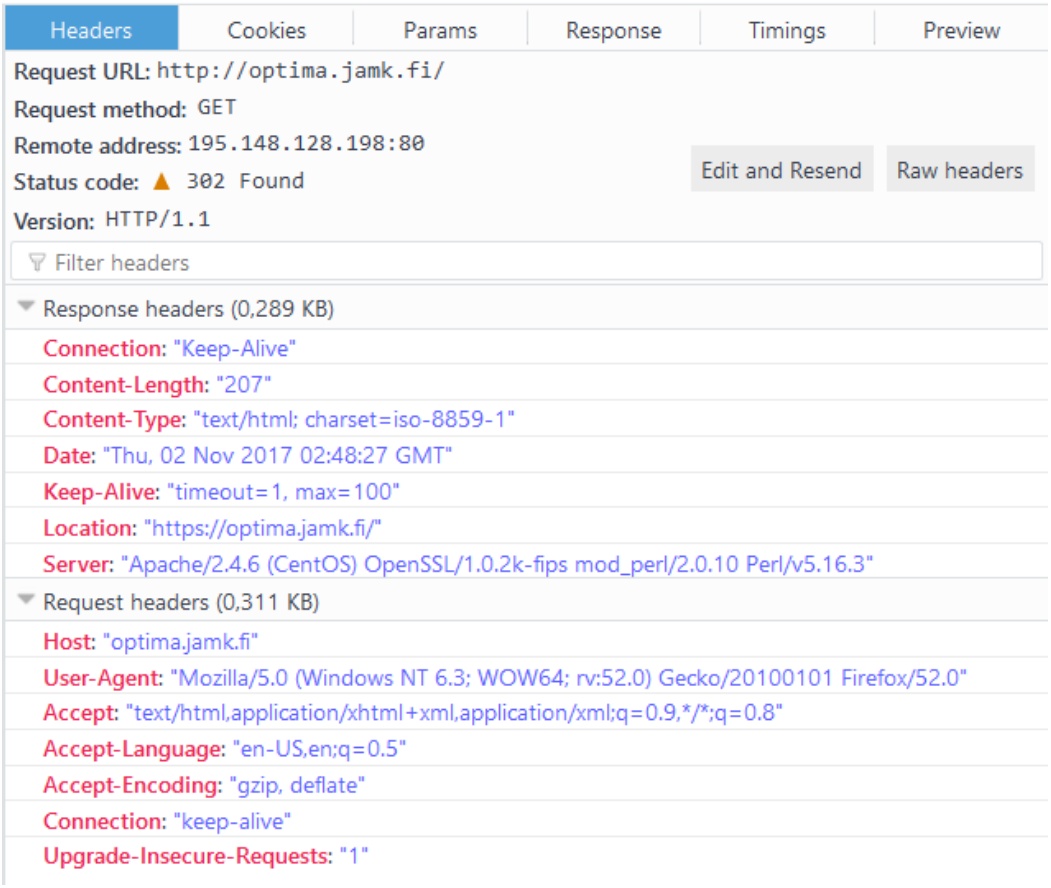
- entity-header. (RFC 2616 1999.)

General-header-otsikko sisältää yleistietoa, jota voidaan käyttää sekä pyyntö- että vastausviesteissä. Entity-header-otsikko sisältää metainformaatiota sivustosta. (RFC 2616 1999.)

Request-header:n avulla käyttäjä voi lähettää lisätietoa serverille tekemästä pyynnöstään tai lisätietoja itsestään. Näitä tietoja ovat esim. käyttäjän selain sekä se mitä tiedon esitysmuotoja käyttäjä sallii. Tärkein ominaisuus on kuitenkin auktorisoinnin mahdollistaminen. (RFC 2616 1999.)

Vastausotsikon avulla palvelin voi lähettää lisätietoja vastausviestissään jotka eivät mahdu status-kenttään. Otsikko sisältää lisätietoja palvelimesta sekä haetusta resurssista. (RFC 2616 1999.)

Kuviossa 2 on esimerkki kahden eri otsikon käytöstä. Kuviossa palvelin kertoo lisätietoja itsestään ja sen lisäksi käyttäjä määrittää sen mitä eri tiedon esitysmuotoja hän tukee.



Kuvio 2 Otsikot HTTP-paketissa

3 Incident management

3.1 Tapahtuma

Tapahtumalla viitataan tietoverkossa tapahtuneisiin havaittuihin asioihin. Näitä asioita voivat olla esim. käyttäjän kirjautuminen tai VOIP-puhelun soittaminen. Jotta tapahtuma voidaan havaita, siitä on oltava merkintä tapahtumienhallinnan ohjelmistossa tai lokitiedostoissa. Tapahtuman havaitseminen ei riitä vaan ilmennyt poikkeustapahtuma pitää myös tiedostaa ihmisen toimesta.

3.2 Poikkeama

Incident eli poikkeustapaus tai toisin sanottuna poikkeama on normaalista poikkeava epäsuotuisa tapahtuma (Johnson 2013, 66).

Määritelmään mukaan jokainen tapahtuma, jolla pyritään aiheuttamaan vahinkoa kohdeyritykselle, on poikkeama. Poikkeama voi esim. tarkoittaa järjestelmään murtautumisyritystä, onnistunutta järjestelmään kirjautumista vieraasta IP-osoitteesta tai palvelunestohyökkäystä. Hyökkäys voi kohdistua tietojärjestelmiin, tietokoneisiin tai yrityksen verkkoon. (Schperberg, Brancik 2005. kpl 4 Incident.)

Taulukossa 4 on kuvattu NIST-organisaation ohjemallissa määritellyt mahdolliset tietoturvauhat, joista selviää, että mitä mahdollisia poikkeamia verkossa voidaan havaita.

Taulukko 4. Tietoturvauhat (Cichonski, Millar, Grance, Scarfone 2012. 25-26.)

Keino	**Menetelmä**
Siirrettävä media	USB-tikun tai kovalevyn kautta suoritettava hyökkäys.
Brute-force	Palvelunestohyökkäys tietoverkkoon tai vaihtoehtoisesti joku voi yrittää brute force -menetelmällä murtaa salauksen.
Sähköposti	Phishing-hyökkäykset, murtautumisyritykset käyttäjätileille.
WWW-palvelut	Web-palveluun hyökkäys voi tapahtua mm. SQL-injektion tai XSS-hyökkäyksen kautta.
Matkimis-yritys	Hyökkääjä luo todelliselta näyttävän palvelun, jota kautta hän pystyy hyökkäämään järjestelmään. Esim. vale WLAN tukiasema, MitM-hyökkäys tai todelliselta näyttävä mock-up-palvelu verkossa.
Materiaali varkaus	Voi ilmentyä esim. tärkeän laitteiston siirtymisenä offline-tilaan, joka voidaan huomata verkonhallinnan ohjelmistosta.
Pääsyoikeuksien rikkominen	Joku yrityksen työntekijä on päässyt käsiksi informaatioon johon hänellä ei ole pääsyoikeutta. Tämä voi olla esim. johtajien jakokansio tiedostopalvelimella. Vaihtoehtoisesti tällä käyttäjällä ei ole vaadittua suojaustasoa dokumenttien lukemiseen.

NIST-ohjemallissa määritetään myös lista käytännön esimerkki-tapauksia poikkeamista, tai poikkeamaa edeltävistä tapahtumista jotka voivat johtaa hyökkäykseen.

- *Webpalvelimen lokeista ilmenee, että joku on käyttänyt haavoittuvuusskanneria*
- *Sähköpostipalvelimen ohjelmistosta on löydetty uusi tietoturvauhka*
- *IDS-palvelin havaitsee, että jokin ylikuormittaa tietokanta-palvelimen ohjelmistoa*
- *Virustorjunta ilmoittaa, että yrityksen tietokoneessa on havaittu virus*
- *Ylläpitäjä löytää tiedoston, jossa on paljon erikoismerkkejä.*
- *Joku on mennyt muuttamaan konfiguraatio-asetuksia, ja tästä kertyy lokimerkintä.*
- *Sovellus raportoi monta peräkkäistä epäonnistunutta kirjautumisyritystä*
- *Verkossa leviää sähköposti, jossa on epäilyttävää sisältöä*
- *Verkon kuormituksessa huomataan poikkeavaa kuormitusta* (Cichonski ym. 2012. 26-27.)

3.3 Poikkeamienhallinta ja sen vaiheet

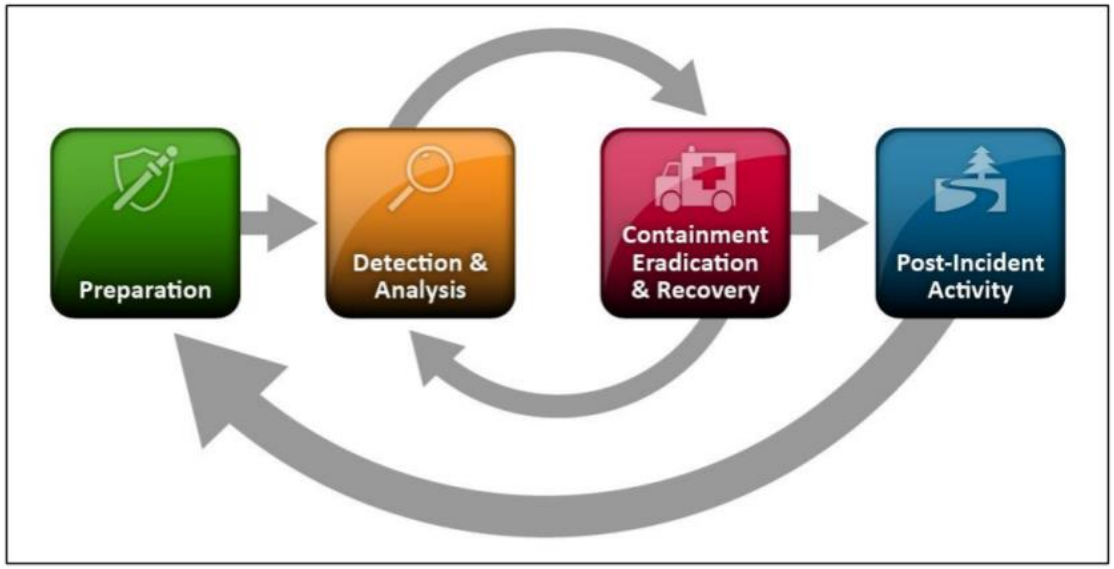
3.3.1 Yleistä

Poikkeustapauksien käsittely sisältää seuraavat toimenpiteet: tapauksen huomiointi, tapaukseen vastaaminen, tapauksesta toipuminen ja jälkitoimenpiteet sekä tapauksen dokumentointi. Poikkeamienhallinnalla pyritään myös varmistamaan yrityksen normaali toimivuus poikkeamien aiheuttamien poikkeustilanteiden aikana. (Kim, Solomon 2014. kpl 8.)

Prosessi vaihtelee eri organisaatiosta tai toimialasta riippuen. NIST on määrittänyt poikkeamienhallinnalle 7 vaihetta. Nämä vaiheet ovat:

- valmistautuminen
- havainnointi ja analysointi
- karanteeni, hävittäminen ja palautuminen
- jälkitoimenpiteet.

Kuten OODA-kierteessäkin, on tämä prosessi jatkuva toimenpide, jota pyritään aina parantamaan. Kuviossa 3 on esitetty NIST-organisaation määrittämä ohjemalli-prosessi poikkeamanhallinnalle. (Cichonski ym. 2012. s. 21.)



Kuvio 3 Poikkeamanhallinnan malli (Cichonski ym. 2012, 21.)

3.3.2 Valmistautuminen (preparation)

Poikkeustilanteisiin voidaan valmistautua kouluttamalla henkilöstöä, luomalla erityistiimi poikkeamienhallintaa varten, määrittämällä vastuualueet sekä ostamalla tarvittavat ohjelmistot ja työkalut. Hyvin luotu tietoturvasuunnitelma vähentää poikkeustilanteita. On siis syytä konfiguroida palomuuri hyvin, kouluttaa henkilöstöä, järjestää lokienhallinta sekä toimiva varmuuskopiointi. (Cichonski ym. 2012, 23.)

Tietoturvatiimin näkökulmasta on syytä tehdä laite- ja ohjelmistokatsaus josta ilmenevät yrityksen laitteisto sekä niissä olevat käyttöjärjestelmät. Tietoverkon tulee olla hyvin dokumentoitu ja kuvattuna graafisesti sekä fyysisenä että loogisena topologiakuvana. Verkosta tulee ottaa baseline-mittaus, joka auttaa luomaan kuvan siitä millainen on normaali verkon kulutus tavallisena päivänä. Tietoverkon portit tulee olla

dokumentoituna, ja yleisimmin käytössä olevien porttien tulee ilmentyä baseline-mittauksessa. Ajan tasalla pysyminen tietoturvauhista parantaa valmistautumiskykyä. (Cichonski ym. 2012, 23.)

3.3.3 Poikkeustapauksen huomiointi (notification)

Ensimmäinen vaihe poikkeamien käsittelyssä on tiedostaa, että kyseessä on poikkeama eikä normaali verkon tapahtuma. Ilmoitus poikkeamasta voi tulla automaattisesti tapahtumienhallinnan ohjelmistosta. Myös virustorjunta tai palomuuri voivat antaa ilmoituksen. Kolmas tapa saada ilmoitus on käyttäjien raportoimana.

Poikkeama on huomioitu, kun siitä on annettu ilmoitus. Poikkeama on tiedostettu vasta silloin kun sitä vastaan aletaan tehdä toimenpiteitä ja tutkimusta. Poikkeaman huomioimisen jälkeen on otettava selvää, että onko kyseessä vakava poikkeama ja että onko se tarkoituksenmukainen mahdollinen hyökkäys. (Kim, Solomon 2014, kpl 8 Steps to take in handling an incident.)

Poikkeaman havainnoinnin jälkeen poikkeama tulee luokitella. Poikkeama ei välttämättä vaadi toimenpiteitä jolloin se voidaan kuitata pois. Poikkeama voidaan luokitella esim. kolmeen tasoon. Nämä ovat alhainen-, keski- ja korkea uhkataso. Uhkat tulisi hoitaa uhkatason perusteella, eikä niiden saapumisjärjestyksessä. (Cichonski ym. 2012. 32.)

Uhkien priorisoinnissa tulee miettiä, että mihinkä osa-alueisiin uhka voi vaikuttaa. Apuna voidaan käyttää CIA-menetelmää, joka tulee sanoista confidentiality, integrity ja availability. Eli siis luotettavuus, eheys ja saatavuus. (Raggad 2010. kpl 1 CIA Triad.)

Saatavuus meinaa sitä, että onko palveluntarjontaan tullut häiriöitä, tai onko palvelu kokonaan poissa käytöstä poikkeaman johdosta. (Raggad 2010. kpl 1 CIA Triad.)

Eheys viittaa siihen, että voidaanko dataan tai informaatioon luottaa. Eli siis onko informaatioon kuten esimerkiksi yrityksen intran web-sivustoon tehty muokkauksia hyökkääjän toimesta. (Raggad 2010. kpl 1 CIA Triad.)

Luotettavuus viittaa siihen, että informaatio pysyy suojattuna ja salattuna. On tärkeätä salata tieto kryptograafisin menetelmin, jolloin pahimmassa tilanteessa vuode-

tut tiedot ovat käyttökelvottomia tehokkaasta salauksesta johtuen. Luotettavuus viittaa myös siihen, että onko tieto pysynyt suojattuna, vai onko joku päässyt käsiksi tietoihin. (Raggad 2010. kpl 1 CIA Triad.)

Muita luokittelutapoja voi olla esim. poikkeamasta palautumisen kestoaika. Eli siis, jos poikkeamaa ei hoideta pois alta nopeasti, se voi tarkoittaa, että paljon vahinkoa voi tapahtua hitaan ratkaisuprosessin aikana. Poikkeamat voidaan myös luokitella niiden käsittelyajan perusteella, ja sen perusteella voidaan miettiä optimaalista resurssienkäyttöä. Monta pienempää uhkaa voidaan ratkaista samassa ajassa, kuin mitä yksi keskisuuri uhka vaatisi. (Cichonski ym. 2012. 33.)

3.3.4 Poikkeustapaukseen vastaaminen (response)

Poikkeamaan vastaaminen tarkoittaa konkreettisten toimenpiteiden aloittamista. Tärkeä ensivaihe on eristää uhka, jottei se pääse leviämään tietoverkossa. Organisaatiolla on hyvä olla luotuna suunnitelma, jonka pohjalta poikkeustapauksiin lähdetään vastaamaan. Suunnitelmasta tulee ilmetä, että mitä toimenpiteitä suoritetaan poikkeaman ilmentyessä. Esimerkiksi jos tietokonevirus on iskenyt yhteen työasemaan, tällöin kyseinen työasema tulee kytkeä pois verkosta, kunnes uhkatilanne on hallinnassa ja virus on saatu poistettua. Hyvin laaditun suunnitelman avulla poikkeustilanteet saadaan ratkaistua nopeasti, ja tällöin myös palveluiden saatavuus turvataan. (Kim, Solomon 2014. kpl 8.)

Poikkeamien ratkaisussa on oleellista selvittää poikkeustilanteen lähde ja alkuperä. Lähde voi siis olla esim. web-palveluun kohdistunut hyökkäys, ja hyökkäyksen alkuperä on kiinalaisesta IP-osoitteesta. Pelkkä web-palvelun tietoturvauhan korjaaminen ei välttämättä riitä, koska hyökkäykset voivat jatkua muilla tavoin samasta alkuperäisestä osoitteesta. Tällöin on syytä esim. muokata palomuurin asetuksia, jolloin hyökkäykset saadaan loppumaan lopullisesti. Kun on selvitetty, että mihinkä hyökkäys kohdistui, voidaan samalla korjata muita kyseiseen palveluun liittyviä tietoturvauhkia ja näin varautua ennalta tuleviin hyökkäyksiin. (Kim, Solomon 2014. kpl 8.)

3.3.5 Poikkeustapauksesta toipuminen (recovery)

Ennen varsinaisia palautumis-toimenpiteitä, on uhka hävitettävä täysin verkosta. Tämä voi tarkoittaa viruksen poistamista, kaapattujen käyttäjätilien sulkemista sekä hyökkäyksen mahdollistavien palveluiden väliaikaista sulkemista.

Poikkeustapauksiin vastaamisen prosessi edellyttää myös vahinkojen korjaamista ja minimointia. Tällöin on tärkeätä, että varmuuskopioinnit ovat ajan tasalla. On selvitettävä, että milloin hyökkäys tapahtui ja mikä on viimeisin varmuuskopiointi hyökkäystä edeltävältä ajalta. Tietoverkossa vahingon voidaan katsoa tulleen korjatuksi silloin, kun tietoverkko on saatu takaisin poikkeustapahtumaa edeltäneeseen tilaan. Yrityksessä taas poikkeustapahtuma voidaan katsoa tulleen korjatuksi silloin, kun yrityksen tuotanto saadaan käyntiin normaalille tasolle. (Kim, Solomon 2014. kpl 8.)

Kun toipuminen on saavutettu, on sen jälkeen tärkeätä vielä kerran analysoida poikkeustapahtuma ja luoda suunnitelma ennaltaehkäiseville toimenpiteille, jotta samaa uhkaa ei olisi enää jatkossa. Jos hyökkäys kohdistui yhteen osaan web-palvelua, on syytä korjata myös muita mahdollisia tietoturvauhkia. Jälkitoimenpiteisiin kuuluu myös henkilöstön kouluttaminen ennalta vastaavien tapahtumien suhteen. (Kim, Solomon 2014. kpl 8.)

Mahdollisia laillisia toimenpiteitä varten tapaus tulee dokumentoida mahdollisimman hyvin. On dokumentoitava, että mihin tietoverkon osa-alueisiin hyökkäys kohdistui, mistä IP-osoitteista hyökkäys tehtiin, mitkä olivat vaikutukset tietoverkon toimintaan sekä mitkä ovat arvioidut rahalliset kustannukset. Myös mahdolliset kirjautumiset tai kirjautumisyritykset tältä ajalta on otettava ylös. (Cichonski ym. 2012. 37.)

3.3.6 Poikkeustapauksen jälkitoimenpiteet (follow-up)

Poikkeustapahtumat on syytä dokumentoida, koska se nopeuttaa vastaavien tapahtumien käsittelyä jatkossa. Vastaavasti voidaan myös päivittää poikkeustilanteisiin vastaamista varten luotua suunnitelmaa eli Incident Response Plan. (Kim, Solomon 2014. kpl 8.)

Poikkeustapahtumasta voidaan luoda kolme erilaista dokumenttia; business impact analysis (BIA), business continuity plan (BCP) sekä disaster recovery plan (DRP). Ensimmäisessä versiossa (BIA), selvitetään poikkeaman vaikutuksien yrityksen toimintaan. Seuraavassa versiossa (BCP), selvitetään kuinka poikkeama on vaikuttanut yrityksen toimintaan palveluiden ylläpidon ja toiminnan jatkuvuuden näkökulmasta. Kolmannessa dokumentissa (DRP), luodaan tälle poikkeustapahtumalle oma suunnitelma tapauksesta toipumista ja palveluiden jälleen-käyttöönottamista varten. (Schperberg, Brancik 2005. kpl 4.)

Incident response plan (IRP) eli poikkeamienhallinnan suunnitelma käy läpi yrityksen luoman suunnitelman poikkeustilanteita varten. Suunnitelmasta käy läpi mm. seuraavat asiat: vahinkojen minimointi ja poikkeaman nopea karanteeni, määritetään vastuualueet henkilöstölle poikkeaman sattuessa, tehokas ja nopea paluu-suunnitelma normaaliin toimintaan sekä myös lista toimenpiteistä joilla voidaan ennaltaehkäistä tulevia uhkatilanteita. Tehokas IRP hoitaa poikkeustilanteet mahdollisimman nopeasti ja tehokkaasti, minimoi aiheutetun vahingon ja sisältää ennaltaehkäiseviä toimenpiteitä. (Johnson 2013. 67.)

3.4 Riskinhallinnan tiimi (CIRT)

Poikkeamien selvittämiseen voidaan perustaa riskinhallinnan tiimi eli englanniksi Computer Incident Response Team (CIRT). Voidaan myös puhua englanniksi Computer Security Incident Response Team (CSIRT)-käsitteestä. CIRT voi olla ennalta luotu tai sitten tiimi luodaan tapauskohtaisesti. Riskinhallinnan tiimi voidaan muodostaa IRP-suunnitelman pohjalta tai sitten riskinhallinnan tiimin toiminnalle tehdään oma suunnitelma. (Gibson, 2015. kpl 15 What is a Computer Incident Response team plan?.)

CIRT-tiimin koko riippuu yrityksen koosta. Hyvin toimiva CIRT tarvitsee montaa eri roolihahmoa joita ovat

- Tiimin johtaja on vastuussa poikkeamien käsittelystä sekä tiimin toiminnasta.
- Tietoturva-asiantuntijoiden pääosaamisalueena ovat palomuurit, IDS-järjestelmät, yleinen tietoturva-osaaminen sekä poikkeamienhallinnan ohjelmistot.
- Tietoverkko-asiantuntijat tuntevat verkon fyysisen sekä loogisen rakenteen, ja myös sen mitä palveluita ja palvelimia verkossa on.

- Vartioinnin henkilökuntaa voidaan tarvita, jos poikkeaman epäillään tapahtuneen talon sisällä. He ovat vastuussa sisäänpääsyn menetelmistä sekä valvontakameroista.
- Lakimiehet tuntevat tapauskohtaiset lakipykälät, ja sen miltä kohdin lakia on rikottu. Heitä tarvitaan myös silloin, jos tapaus etenee oikeuteen asti
- Viestinnän henkilöstöä voidaan tarvita, jos tapauksesta tulee ilmoittaa asiakasyrityksille, tai medialle.
- Henkilöstön hallinta tuntee yrityksen rakenteen sekä voi tarvittaessa ottaa selvää työntekijöiden yhteystiedoista (Gibson, 2015. kpl 15 Elements of a CIRT Plan.)

Tiimin toimivuus sekä tapauksien onnistunut ratkaiseminen tarvitsevat toimiakseen hyvän kommunikoinnin eri henkilöstön välillä. Yrityksellä voi olla yksi tiimi joka hoitaa kaikki poikkeustapaukset, tai monta eri tiimiä joilla on eri vastuualueet. Osa tiimin jäsenistä voidaan ulkoistaa, kuten esim. lakimiehen palvelut. CIRT-tiimin toiminta voidaan myös kokonaan ulkoistaa. Yrityksen on myös päätettävä, että millä ajanjaksolla CIRT-tiimi on käytettävissä. Korkean tietoturvatason yritys voi tarvita ympäri vuorokautisen CIRT-tiimin. (Cichonski ym. 2012 14-15.)

3.5 Tilannetietoisuus

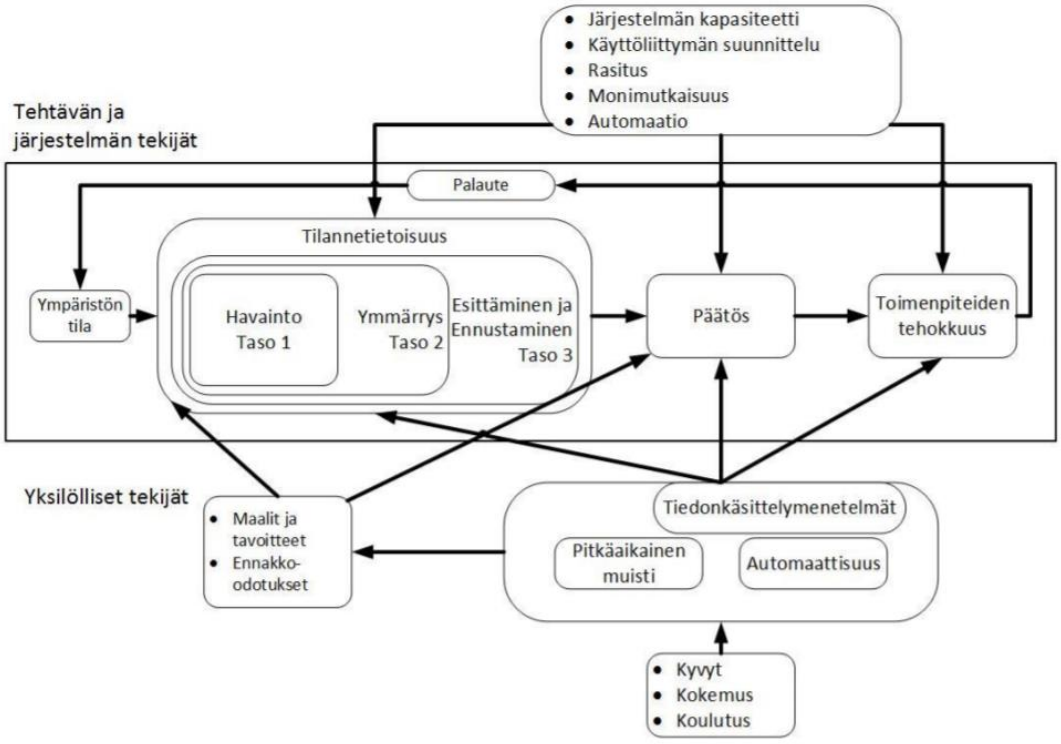
Tietoisuus voidaan määrittää kykynä havainnoida poikkeustilanteet normaaleina pidetyistä tilanteista. Se merkitsee myös kykyä määrittää uusien tilanteiden taso suhteessa normaaliksi määritettyyn tasoon. Tilannetietoisuus merkitsee kirjaimellisesti kykyä olla tietoinen tilanteesta, kunhan tilanteelle on ensin määritetty normaalitaso, jonka kautta voidaan myös tunnistaa poikkeamat. Puhutaan myös normaalista ja poikkeavasta käytöksestä. Käytös voi tietoverkoissa tarkoittaa esim. käyttäjien toimenpiteitä, palvelimen resurssien kulutusta tai erilaisia sisäänkirjautumisia järjestelmiin. (Amoroso 2011. kpl 1 awareness.)

3.6 Endsleyn malli

Mica Endsley (1995) on kehittänyt oman mallinsa tilannetietoisuudelle. Endsley pyrkii mallillaan tehostamaan johtopäätöksien sekä tulevaisuuden arvioinnin tärkeyttä. Hänen mukaansa tilannetietoisuus ei voi vain koostua havainnoinnin vaiheen tärkeydestä. (Endsley 1995. 1.)

Endsley painottaa mallissaan myös ihmisen osuutta osana tilannetietoisuuden mallia. Ihmiselementit koostuvat lyhytkestoisesta muistista, havainnointikyvystä, työmuistista sekä pitkäkestoisesta muistista. Endsleyn mukaan ihmisen tarkkaavaisuus on rajallista, mikä taas luo haasteita havainnointitasolle. Endsleyn mukaan ihmisen havainnointikykyyn vaikuttavat myös pitkäkestoinen muisti sekä ennalta opitut arvot ja asenteet. Esimerkiksi jos ihminen tuntee ympäristönsä hyvin kokemuksensa pohjalta, tällöin hänen huomionsa keskittyy uusiin asioihin ympäristössä, mikä taas johtaa tehokkaampaan tilanteiden havaitsemiseen. (Endsley 1995, 42-43.)

Endsleyn malli on esitetty kuviossa 4.



Kuvio 4 Endsleyn tilannetietoisuus-malli (Heimonen 2017, 49.)

Endsleyn malli koostuu kolmesta perusosasta

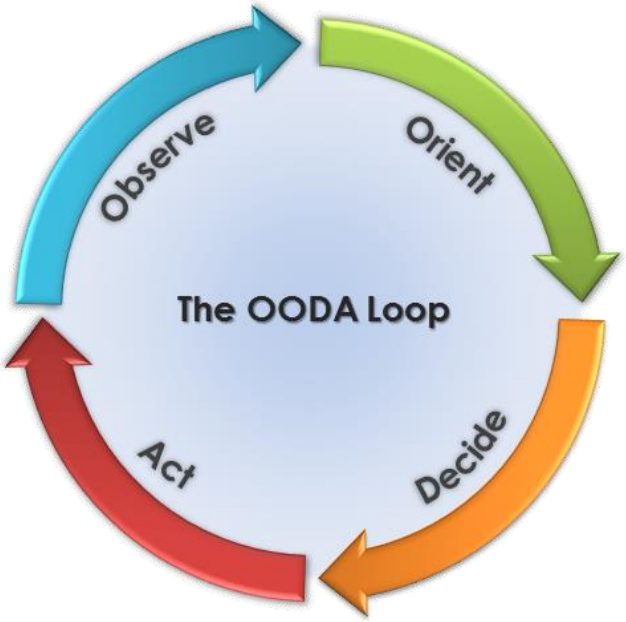
- Havainnointi-vaihe koostuu ympäristön tarkkailemisesta. Tässä vaiheessa kiinnitetään huomiota ympäristön kannalta tärkeisiin kohteisiin. Kohteissa kiinnitetään huomioita niiden yleistilaan, kohdetta kuvaaviin tekijöihin sekä dynaamisiin vuorovaikutustekijöihin.
- Ymmärrys-vaiheessa aletaan muodostaa johtopäätöksiä edellä tehdyistä havainnoista. Havainnoijan pitää olla tietoinen mitenkä eri tekijät ovat vuorovaikutuksessa keskenään, sekä millaisia muutoksia ympäristöön tietyt tapahtumat voivat aiheuttaa.
- Kolmannessa vaiheessa pyritään ennustamaan tulevaisuutta ensimmäisessä vaiheessa tehtyjen havaintojen perusteella, sekä toisessa vaiheessa tehtyjen johtopäätöksien ja päätelmien perusteella. Arvioimalla tulevaisuuden näkymiä, pyritään luomaan mahdollisimman tehokkaita päätöksiä nykyhetkessä, joilla on paras mahdollinen arvioitu vaikutus tulevaisuudessa. (Endsley 1995, 36-37.)

3.7 OODA-loop

OODA-loop eli suomennettuna OODA-kierre tulee sanoista Observe, Orient, Decide ja Act. Suomennettuna nämä sanat ovat havainnoi, määrittele, päätä ja toimi. Menetelmän kehitti Yhdysvaltain armeijan eversti John Bloyd. Menetelmää käytettiin alun perin maavoimien operaatioissa. (Taylor 2012. kpl 9 The OODA Loop.)

Bloyd on sanonut pitävänsä toista vaihetta eli määrittelemis-vaihetta kaikista tärkeimpänä. Määrittelyvaihe on jatkoa havainnoinnille, ja se on edeltävä vaihe ennen päätöksentekoa. Havaintoja voi olla useita, ja ne eivät välttämättä poikkea normaalista. Määrittelemällä havainnot voidaan päättää tarvitseeko toimia. (Taylor 2012. kpl 9.)

Kuviossa 5 on kuvattuna OODA-kierteen toimintaKuvio 5. OODA-kierteen tarkoituksena on jatkuva päätöksenteon kehittäminen, joka perustuu uuden informaation vastaanottamiseen, ja sen pohjalta parannettuihin jatkotoimenpiteisiin. (Keanini 2014.)



Kuvio 5 OODA-kierre (Keanini 2014)

3.8 Cyber Kill Chain

”Kill Chain” on armeijan käyttämä termi, jolla kuvataan hyökkäyksen kaikki eri vaiheet kohteen tunnistamisesta kohteen tuhoamiseen saakka.

Yhdysvaltalainen puolustusalan yritys Lockheed Martin on kehittänyt oman versionsa tästä mallista kyberturvallisuudelle. Mallilla pyritään tunnistamaan ja välttämään kyberhyökkäykset. Mallin eri vaiheet havainnollistavat hyökkäyksen kulun. Hyökkääjän pitää selviytyä jokaisesta vaiheesta, jotta hyökkäys lopulta onnistuu. (Gaining the advantage 2015.)

Malli koostuu 7. eri vaiheesta, jotka ovat esitettynä taulukossa 5.

Taulukko 5. Cyber Kill Chain -vaiheet (Gaining the advantage 2015.)

Vaihe	**Toimenpiteet**
Tiedustelu (Reconnaissance)	Tiedustellaan kohdehenkilöitä ja kerätään heidän sähköpostiosoitteitaan sekä tehdään taustatutkimusta yrityksestä. Lisäksi koitetaan etsiä yrityksen palvelimia verkosta.
Aseistaminen (Weaponization)	Luodaan haittaohjelma, joka on tyypillisesti automatisoitu ”backdoor”-virus.
Jakelu (Delivery)	Levitetään virusta kohteeseen, esim. saastuneiden USB-tikkujen ja sähköpostiviestien avulla
Hyödyntäminen (Exploitation)	Hyödynnetään tietoturva-aukkoa, jonka avulla virus pääsee levittäytymään kohteessa.
Asentaminen (Installation)	Asennetaan haittaohjelmia kohteeseen.
C2-jakeluverkko (Command & Control)	Kohdelaite liitetään C2-jakeluverkkoon, jolloin sitä laitetta voidaan operoida etäkäytön avulla.
Lopullisten tavoitteiden toteuttaminen (Action of objectives)	Nyt kun on suora yhteys laitteeseen, voidaan alkuperäiset tavoitteet toteuttaa kohdeverkossa. Näitä voivat olla mm.: käyttäjätietojen kerääminen, käyttöoikeuksien muuttaminen, tiedon kerääminen sekä kohdeyrityksen saastuttaminen entisestään uusilla haittaohjelmilla.

4 Cyphon

4.1 Yleistä

Cyphon on avoimen lähdekoodin ohjelmisto poikkeamienhallinnalle. Sen on kehittänyt ja sitä hallitsee Dunbar-niminen yritys.

Cyphon kerää analysoitavaa dataa monesta eri lähteestä. Näitä lähteitä ovat: sähköposti, lokitiedostot sekä API:t. Tietoa voidaan myös hakea sosiaalisesta mediasta, joka rajoittuu vain twitteriin tällä hetkellä. (Overview 2017.)

Cyphon kerää tietoa twitteristä twitterin tarjoaman ”Twitter Public Streams API” -rajapinnan kautta. Haku perustuu täsmäsanoihin, maantieteelliseen sijaintiin sekä ad hoc-parametreihin. (Overview 2017.)

Cyphonin avulla voidaan luoda kustomoituja hälytyksiä. Hälytyksiä ja aiempia poikkeustapahtumia voidaan kerätä yhteen, eli niitä kategorisoidaan. Poikkeustapahtumille voidaan asettaa eri hälytysarvoja. (Overview 2017.)

Cyphon-ohjelmisto koostuu taustalla pyörivästä ”Cyphon Engine” -palvelusta, joka hoitaa datan prosessoinnin, sekä front-end käyttöliittymästä nimeltä Cyplops. Vaikka Cyphon on muuten avoimen lähdekoodin ohjelmisto, Cyplops:in lisenssi ei ole kaupalliseen käyttöön tarkoitettu. (Overview 2017.)

4.2 Asennus ja käyttöönotto

Cyphon tarvitsee toimiakseen monta muuta avoimen lähdekoodin ohjelmistoa. Taulukossa 6 on listattuna nämä ohjelmistot sekä niiden käyttötarkoitus.

Taulukko 6. Cyphon-ohjelmiston riippuvuudet

Ohjelmisto	**Käyttötarkoitus**
PostgreSQL	PostgreSQL on relaatiotietokanta.
RabbitMQ	RabbitMQ on viestinvälitys-ohjelmisto. Sen avulla voidaan mm. kuljettaa ja säilöä viestejä, sekä muokata niiden esitystapaa.
Logstash	Logstash kerää lokitiedostoja monesta eri paikasta ja sen jälkeen säilöö ne käyttäjän määrittämään paikkaan. Logstashia voidaan myös käyttää datan käsittelyssä, sekä datan parserina.
Elasticsearch ja/tai MongoDB	Elasticsearch on hakukone, jota voidaan käyttää tiedostoihin. MongoDB on tietokantasovellus, joka ei käytä SQL-kieltä. MongoDB käyttää JSON-syntaksin kaltaista rakennetta.
Nginx tai Apache	Nämä ovat WWW-palvelimia.

Cyphon voidaan asentaa joko manuaalisesti asentamalla kaikki edellä mainitut riippuvuudet tai vaihtoehtoisesti Cyphon voidaan asentaa docker:n avulla. Työssä Cyphon asennetaan docker:n avulla.

Docker:ia varten asennetaan 2 ohjelmistoa jotka ovat Docker Compose ja Docker Community Edition.

Cyphon tarjoaa 2 eri ympäristöä jotka ovat tuotantoympäristö sekä kehitysympäristö.

Komentoriviltä ajettava yksi asennuskomento asentaa kaikki tarvittavat riippuvuudet. Asentamisessa ja lataamisessa kestää kauan aikaa.

Cyphonin käyttäjätunnus sekä käyttäjätunnukset Cyphonin tarvitsemiin apuohjelmistoihin löytyvät ”/opt/cyphon/cyphondock/config/env/cyphon.env”-tiedostosta. Uusi admin-käyttäjätunnus luodaan komentoriviltä.

Cyphonin etusivulla sijaitseva karttapalvelu vaatii rekisteröitymisen Mapbox-verkkopalveluun. Tämän jälkeen mapbox access token lisätään konfigurointi-tiedostoon.

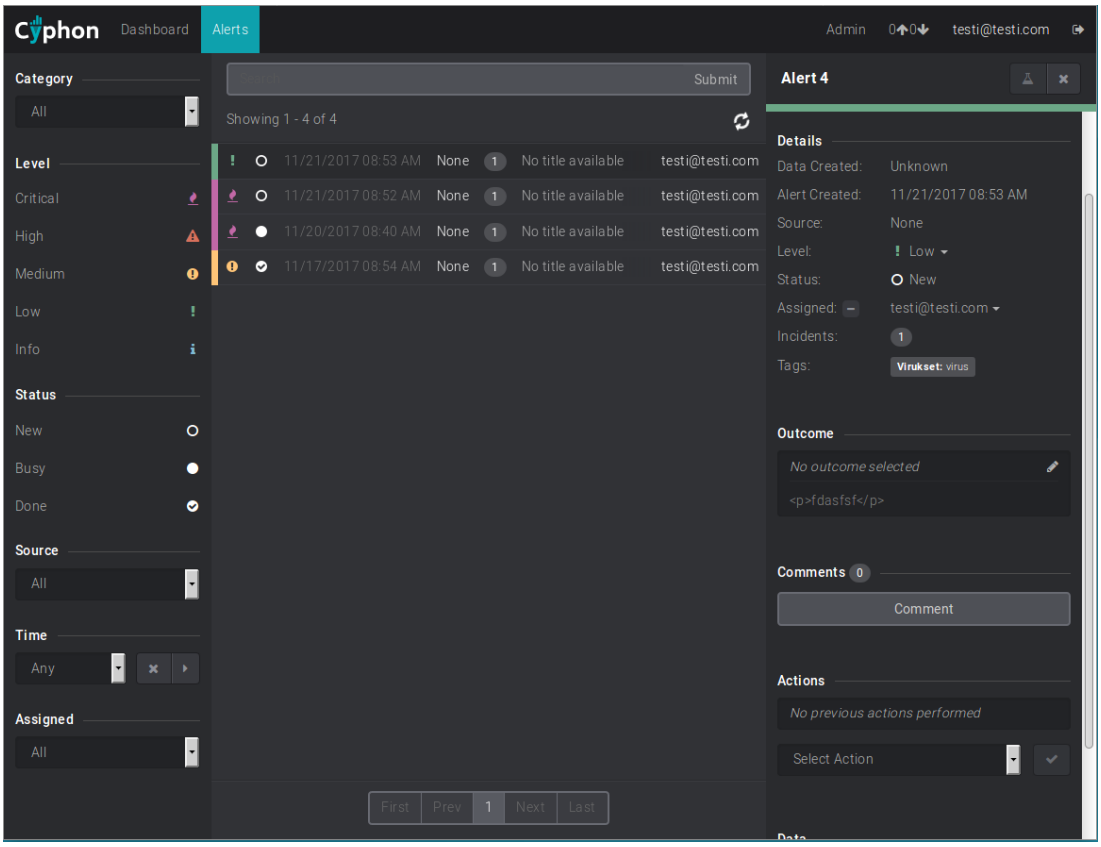
4.3 Ominaisuudet ja käyttöliittymä

Cyphonin aloitussivuna toimii Cyclops front-end GUI, joka on esitettynä kuviossa 6.

Sitä kautta saa yleistietoja meneillään olevista hälytyksistä. Jos hälytyksille on määritelty sijainti niin silloin ne näkyvät kartalla.

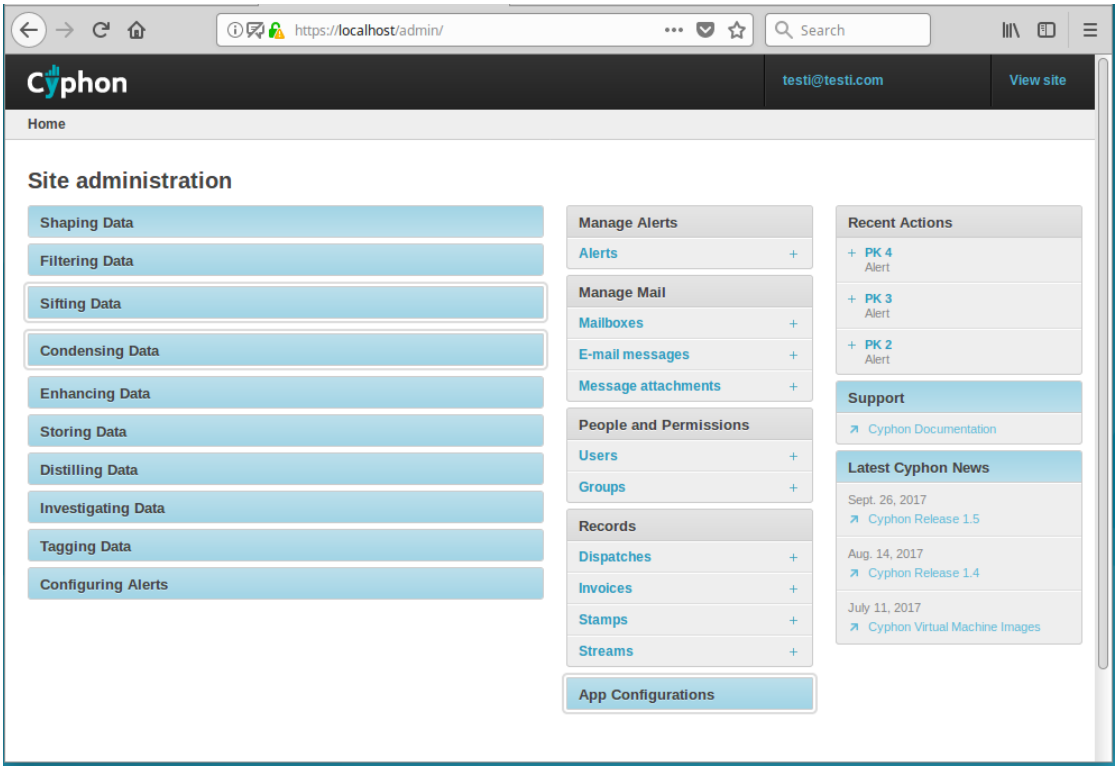
Kuvio 6 Cyclops-käyttöliittymä

Kuviossa 7 on esitettynä Alerts-välilehti, jolta saadaan yksityiskohtaisempaa tietoa hälytyksistä.



Kuvio 7 Alerts-välilehti

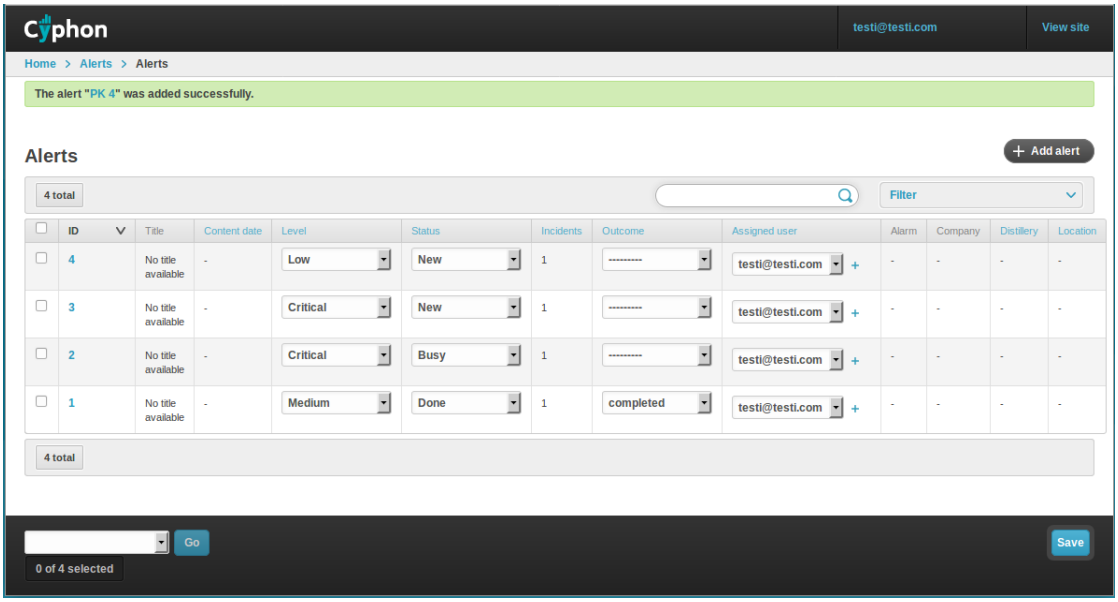
Cyphonia hallinnoidaan admin-paneelista. (Ks. Kuvio 8.) Admin-paneelin kautta voidaan lisätä hälytyksiä, luoda käyttäjiä ja ryhmiä, hallinnoida tageja, määrittää lähteitä automaattiselle tiedonhaulle kuten lokit, sekä määrittää niitä asetuksia joilla data filteröidään.



Kuvio 8 Cyphonin admin-paneeli

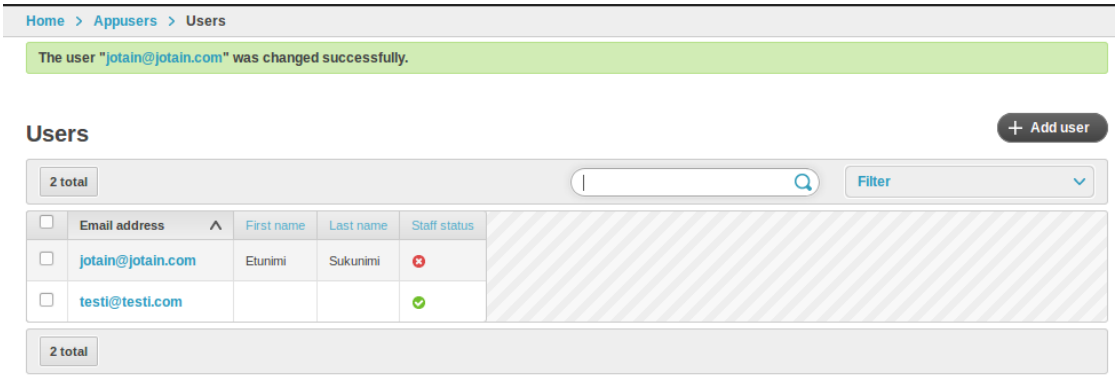
Cyphonin admin-paneeli on hyvin dynaaminen. Käyttäjän ei tarvitse koskaan siirtyä poispäin tietyltä sivulta. Jos esim. puuttuu ryhmä, tarjoaa Cyphon pikalinkin ryhmän lisäämiseen. Tämä vähentää liikkumista sivujen välillä. Tyyliasultaan Cyphon noudattaa yksinkertaista ja suoraviivaista rakennetta.

Hälytyksiä pääsee lisäämään admin-paneelin kautta. Kuvassa 9 on esitettynä Cyphonin admin-paneelin listaus kaikista hälytyksistä.



Kuvio 9 Cyphonin hälytykset

Käyttöoikeuksien hallitseminen on Cyphonissa paljon monipuolisempaa kuin YETI:ssä. Käyttäjälle voidaan määrittää ”staff status” joka kertoo, että kuuluuko hän hallinnoivaan henkilökuntaan. Käyttäjille voidaan määrittää ryhmiä ja ryhmille erikseen käyttöoikeuksia. Myös käyttäjille voidaan määrittää tapauskohtaisesti eri käyttöoikeuksia. Kuviossa 10 on esitettynä Cyphonin käyttäjienhallinta.



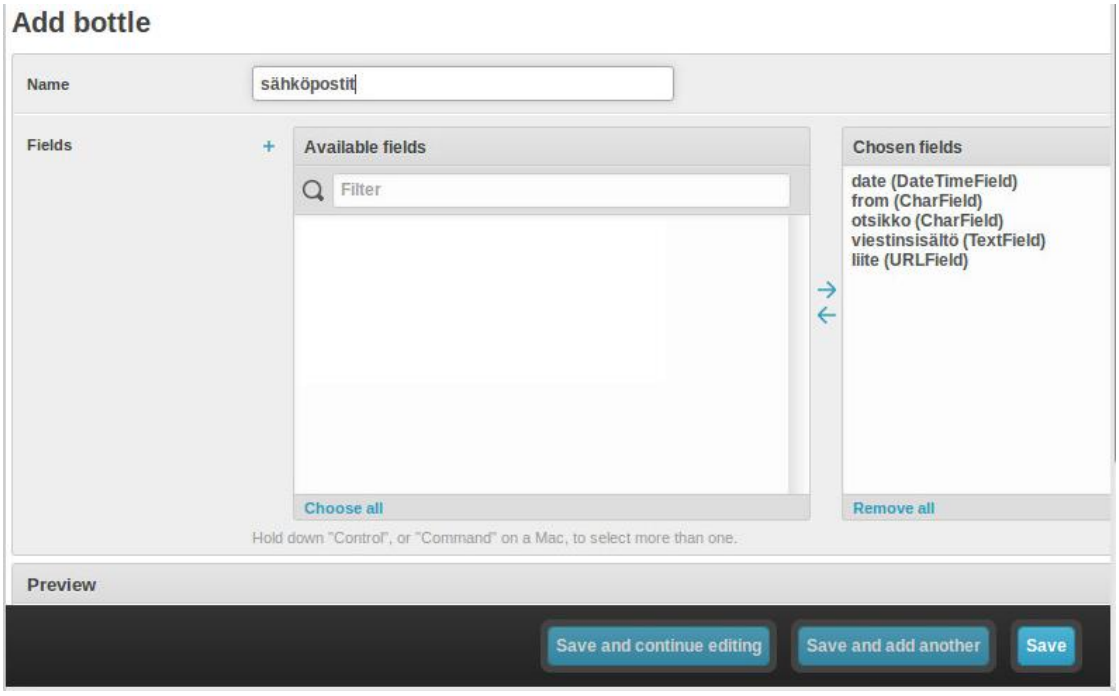
Kuvio 10 Cyphon käyttäjienhallinta

Cyphon on hyvin mekaaninen käyttöliittymältään. Ohjelmisto vaatii paljon opettelua ja siihen totuttelua. Käyttöönottamisen kynnys on suurempi ja vaatii myös enemmän kouluttamista ja ohjelman käyttöön sitoutumista. Data prosessoidaan automaattisesti, joten käyttäjän tulee olla perillä siitä, että millä säännöillä prosessointi tapahtuu.

4.4 Cyphon tiedonkäsittelyn prosessi

4.4.1 Tiedonkäsittelyn perusteet

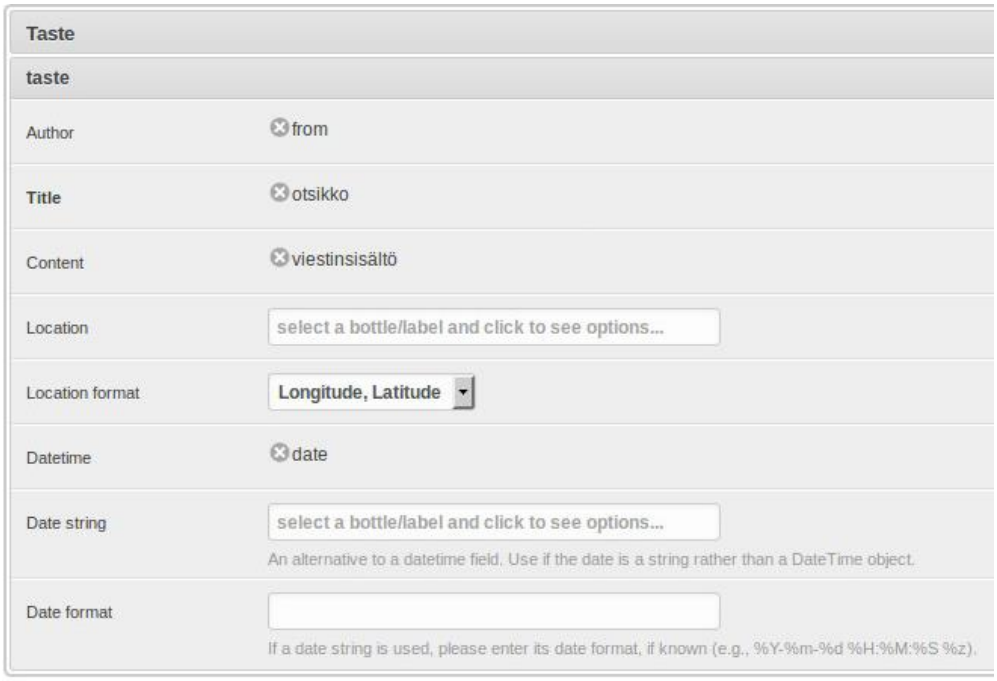
Kerätty tieto tallennetaan Cyphonissa pulloihin. Pullon rakenne kuvaa sitä, että millaista tietoa data sisältää. Pullo koostuu kentistä. Kentille voidaan määrittää monta eri tietotyyppiä. ”Target type” määrittää millainen kentän kohde on kyseessä. Se voi olla käyttäjätili, aikaleima, IP-osoite, avainsana tai sijainti. Kuviossa 11 on esimerkki pullon luomisesti sähköpostiviestien säilömistä varten.



Kuvio 11 Pullon lisääminen Cyphoniin

Pulloille luodaan säiliö. Säiliöille voidaan myös asettaa etiketti tai toisin sanoen leima, jonka käyttäjä voi määrittää kertomaan, että millaista tietoa säiliö sisältää. Säiliöille voidaan antaa oma maku. Maku kuvaa sitä, että millaista sisältöä säiliössä on. Cyphonissa käyttäjä ei voi luoda otsikkoa hälytyksille. Hälytysten otsikko luodaan automaattisesti sen jälkeen, kun ohjelma on määrittänyt maun perusteella, että minkä tyyppistä data on. (Data storage 2017.)

Kuviossa 12 on esimerkki makujen lisäämisestä. Tiedot haetaan edellä luotujen kenttien perusteella, ja tämän avulla Cyphon osaa luoda hälytykseen otsikkokentät.



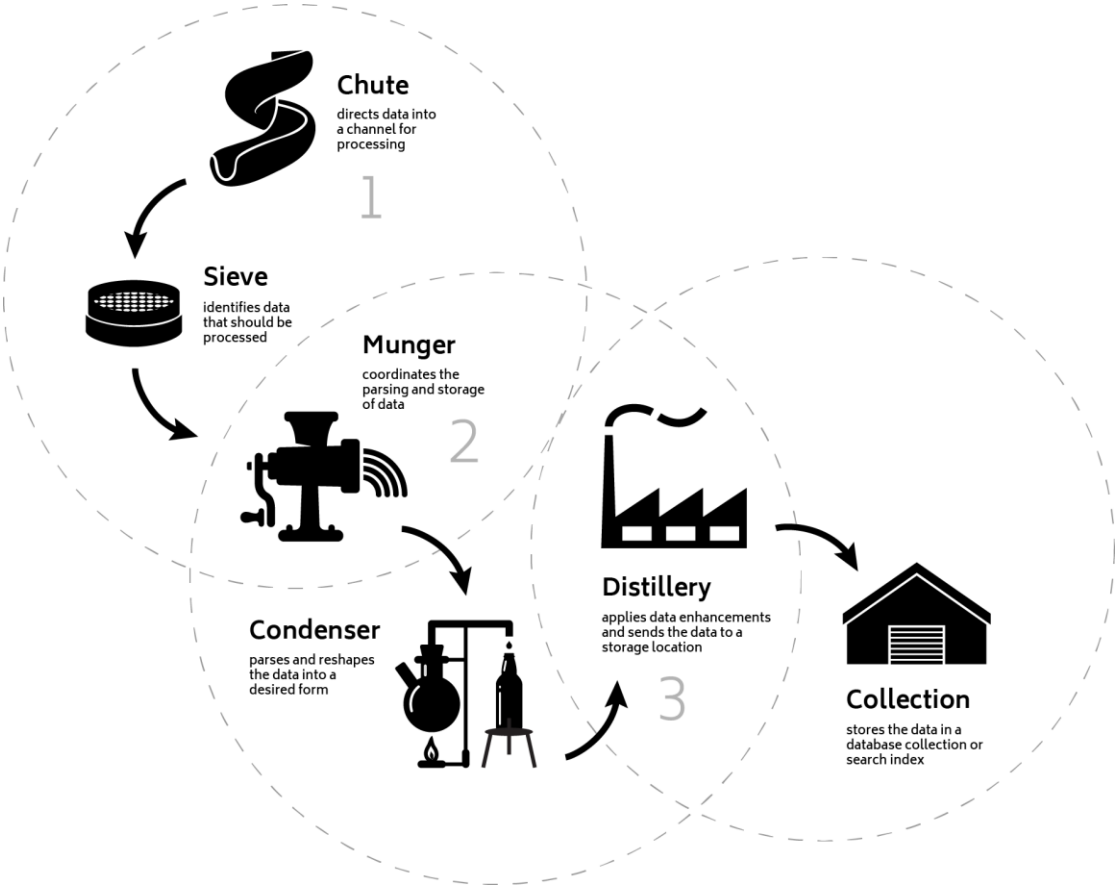
Kuvio 12 Taste-kenttien lisääminen Cyphoniin

Säiliöt tallennetaan varastoihin. Varastot ovat joko Elasticsearch- tai MondoDB-tietokannan pohjaisia. Varasto sisältää kokoelmia. Jokainen kokoelma sisältää vain sille ominaista dataa.

4.4.2 Tiedon prosessoinnin malli

Tieto käsitellään Cyphonissa 5 eri vaiheen kautta joilla on kemiasta lainattu nimi. Nämä vaiheet ovat: kanava, seula, siirtäjä, tiivistin sekä tislaamo. (Data processing 2017.)

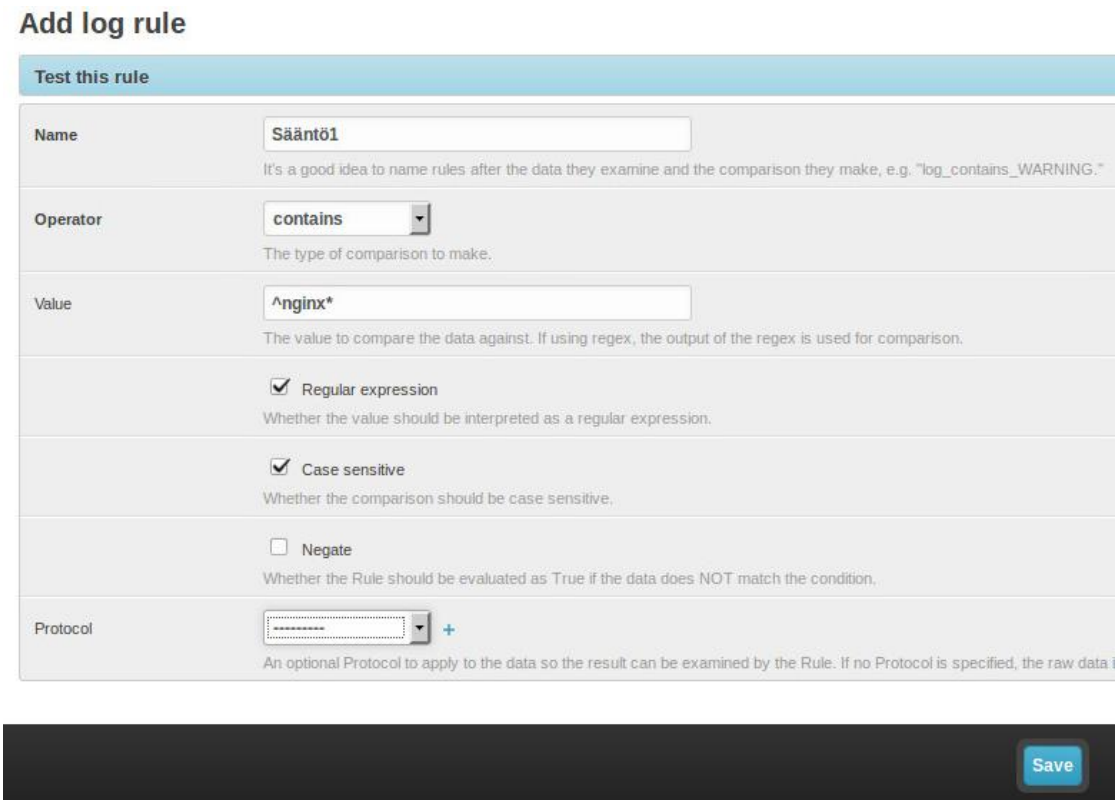
Kuviossa 13 on havainnollistettu tämä prosessi.



Kuvio 13 Cyphonin tiedonkäsittelyn vaiheet (Data processing 2017.)

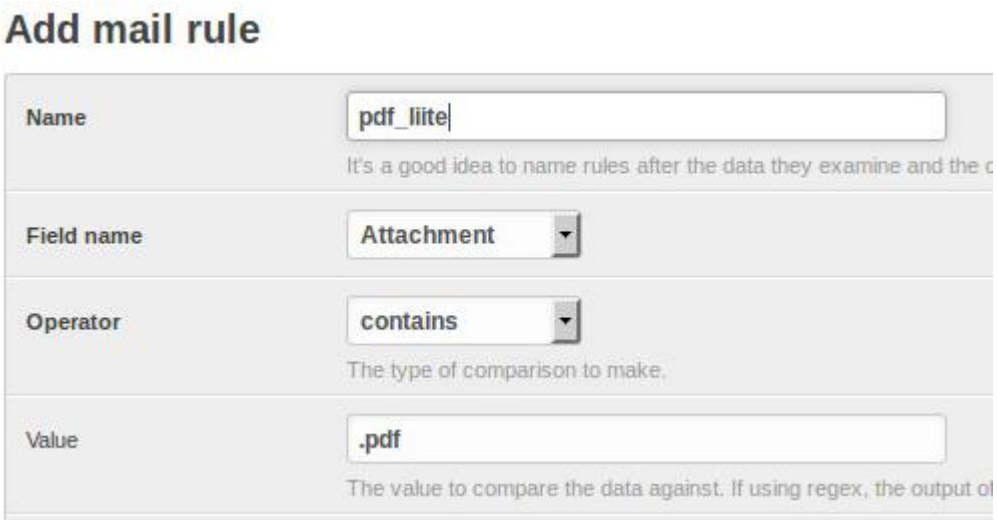
Data saapuu kanavaa pitkin. Seula määrittää, että onko data oikean tyyppistä. Väärän tyyppinen data hylätään.

Kuviossa 14 on luotu seulalle sääntö.
Säännöllä varmistetaan, että kyseinen lokitiedosto sisältää nginx-palveluun liittyvä dataa. Operaattoriksi voidaan asettaa arvot ”contains”, ”begins with”, ”ends with” tai ”equals”. Seulottavalle arvolle voidaan asettaa regex-sääntö. ”Case sensitive” määrittää että pitääkö sanan noudattaa tiettyä kirjoitusasua kirjainkoon suhteen.



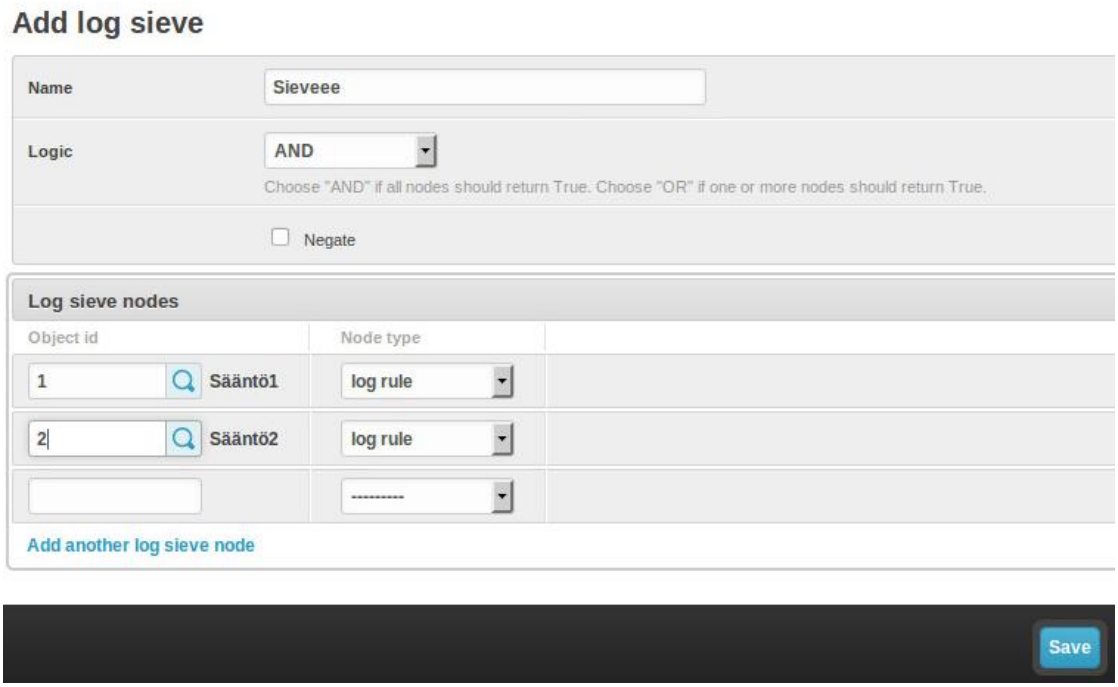
Kuvio 14 Seulan sääntö

Seulalla voidaan myös esim. tarkistaa että onko sähköpostin liitteessä .pdf-tiedostoa. (Ks. Kuvio 15.)



Kuvio 15 Liitteen tarkistaminen

Kuviossa 16 luodaan seula. Logiikka-operaattoriksi voidaan asettaa joko AND, OR tai näiden negaatio.

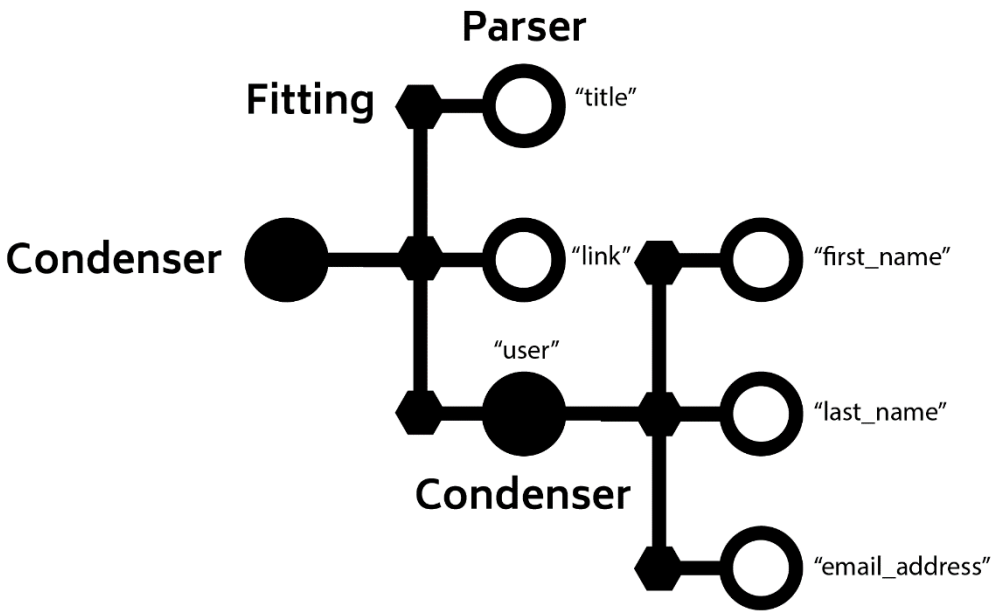


Kuvio 16 Seulan luominen

Seulan jälkeen tieto siirtyy munger:iin, eli siirtäjään. Siirtäjä koordinoi tiedon siirtämisvaihetta. Siirtäjälle määritetään tislaamo sekä tiivistin. Nämä ovat datan säilömis- ja uudelleenmuokkaamisvaiheet.

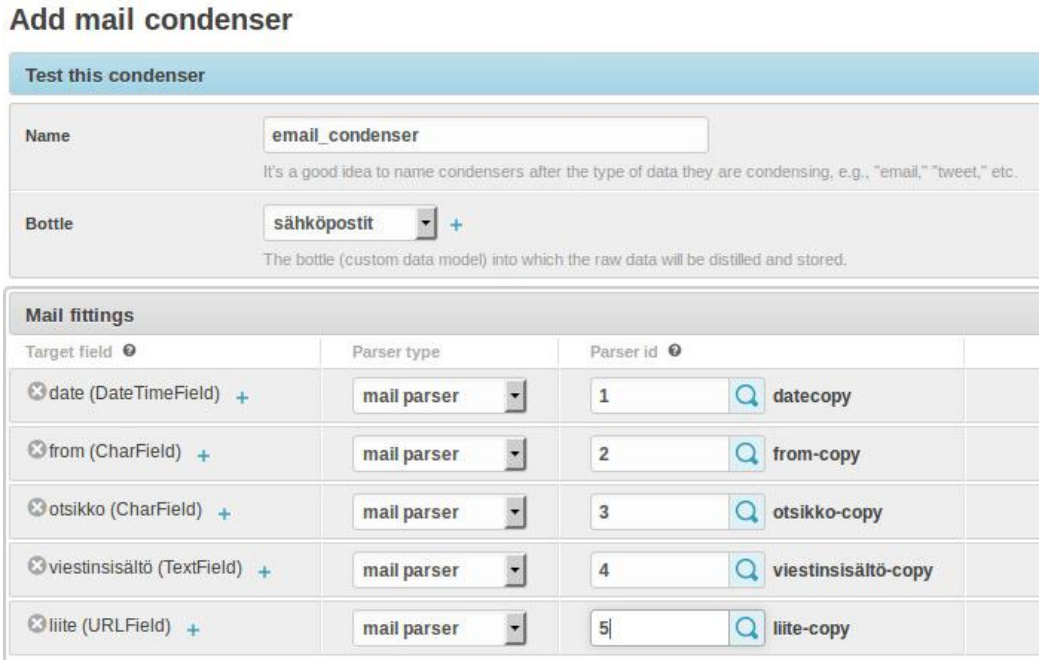
Tiivistintä käytetään siirtämään raaka data uusiin sille luotuihin arvoihin. Arvo joko sopii tai ei sovi. Jos arvo sopii, se siirtyy uuteen ennalta määritettyyn tietotyyppiin parserin kautta.

Kuviossa 17 on esitettynä tiivistimen periaate. Tieto siirretään parserin avulla uusiin ”title”, ”link” ja ”user”-tietotyyppeihin. User-kentän arvot luodaan käyttämällä toista sisäkkäistä tiivistintä.



Kuvio 17 Tiivistimen toimintaperiaate (Data processing 2017.)

Kuviossa 18 on esimerkki tiivistimen käytöstä. Kuvassa luodaan kopiot alkuperäisistä kentistä, mutta vastaavasti aivan eri kentän sisältö voitaisiin siirtää uuteen luotuun kenttään.



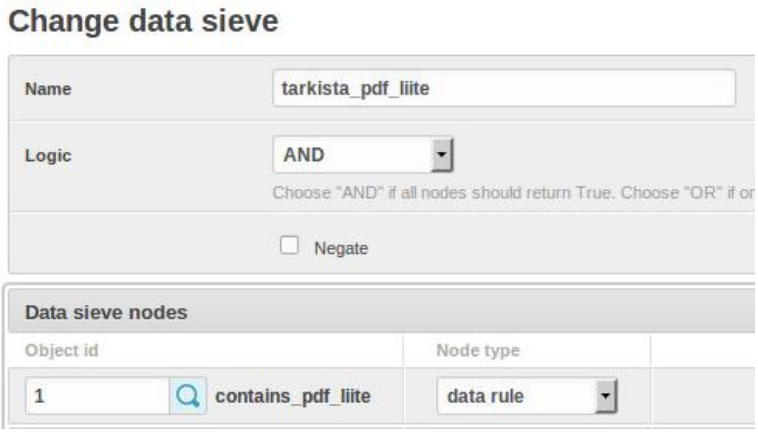
Kuvio 18 Tiivistimen luominen

Viimeisessä vaiheessa eli tislaamossa tiedolle voidaan luoda ehostuksia ja tieto siirretään varastoon. Tislaamolle määritetään varastossa sijaitseva kokoelma ja säiliö. Tiedon ehostaminen tapahtuu joko tarkastamalla data (inspection) tai määritetyllä toimenpiteellä (procedure).

Automaattiset hälytykset saadaan käyttöön konfiguroimalla esim. watchdog. Ensiksi lisätään ”Data rule” joka löytyy JSON Data -valikosta pääkäyttöliittymästä. (Ks. Kuvio 19.)

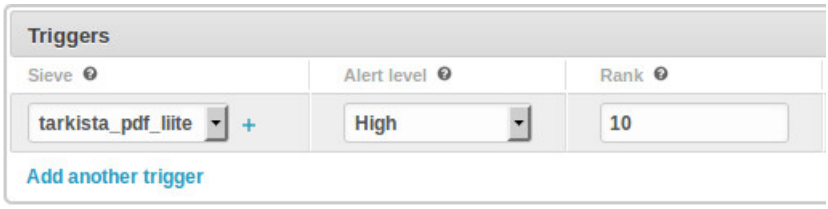
Kuvio 19 PDF-liitteen tarkistus

Luotua sääntöä varten lisätään seula. Seulaan voidaan lisätä monia eri tarkistusvaihtoehtoja. (Ks. Kuvio 20.)



Kuvio 20 Seulan lisääminen

Lopulta watchdog-toiminto otetaan käyttöön ”Configuring alerts” -valikosta. Voidaan myös asettaa muita hälytysvaihtoehtoja. Watchdog-toiminnon tarkastukset prosessoidaan rank-arvon mukaan pienimmästä suurimpaan. (Ks. Kuvio 21.)

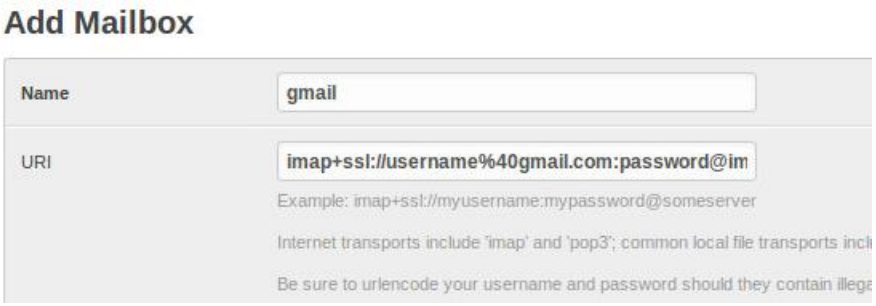


Kuvio 21 Watchdog-toiminnon käyttöönotto

4.5 Tiedon kerääminen

Cyphon kerää tietoa kolmesta eri lähteestä. Nämä ovat sähköposti, lokitiedostot sekä sosiaalinen media joka rajoittuu vain twitteriin tällä hetkellä.

Sähköposteja kerätään automaattisesti määrittämällä sähköpostilaatikko, josta tiedot kerätään. Kuviossa 22 luodaan sähköpostilaatikko Cyphoniin gmail-käyttäjätunnukselle username@gmail.com käyttäen salasanaa ”password”.



Kuvio 22 Sähkopostilaatikon lisääminen

Koko URI-linkki on seuraava:

imap+ssl://username%40gmail.com:password@imap.gmail.com?archive=Archived

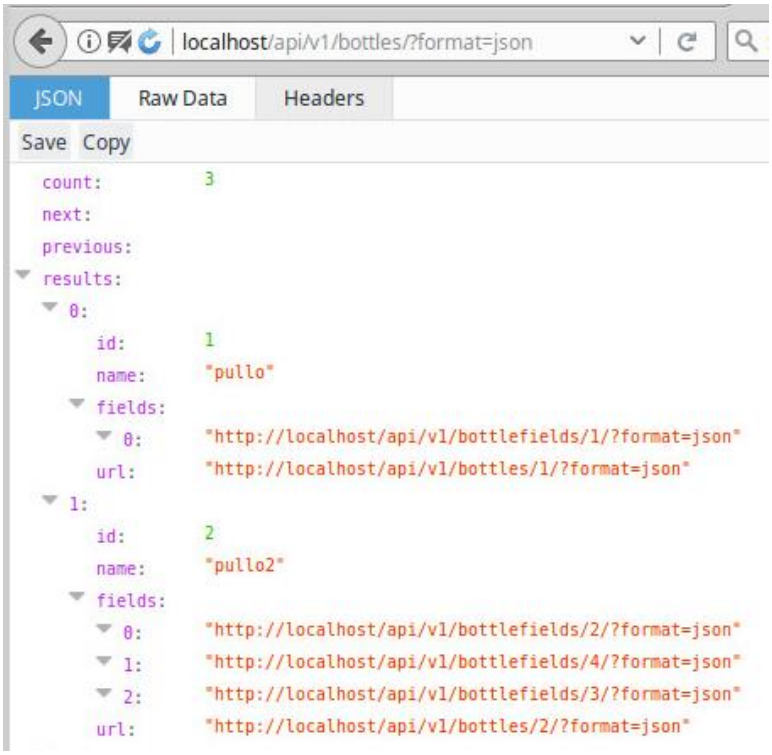
”Archived” meinaa sitä, että haetut viestit arkistoidaan gmail-kansioon.

4.6 API

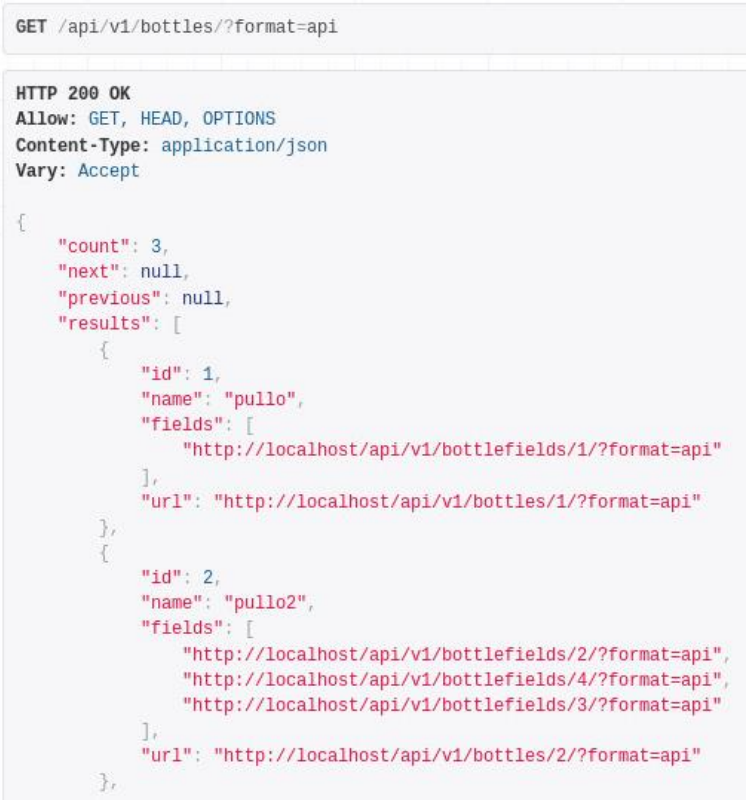
Cyphonin API:n kattava dokumentaatio löytyy osoitteesta: ”localhost:5000/docs/”. API-rajapinnan kautta voidaan hakea lähes jokaista tietokannassa olevaa muuttujaa.

Tiedot saa listattua JSON- tai API-esitysmuodossa. (Ks. Kuvio 23.)(Ks. Kuvio 24.)

JSON-muotoilu on ns. raakaa dataa. API-muodossa hyperlinkit toimivat ja tällöin on helppo selata eri kohteiden välillä.



Kuvio 23 JSON-muotoilu



Kuvio 24 API-muotoilu

Hälytyksistä saa kerättyä helposti perustiedot. (Ks. Kuvio 25.)

```
"results": [
    {
        "id": 4,
        "assigned_user": {
            "id": 1,
            "email": "testi@testi.com",
            "first_name": "",
            "last_name": "",
            "is_staff": true,
            "company": null
        },
        "content_date": null,
        "created_date": "2017-11-24T06:42:23.482684Z",
        "distillery": null,
        "incidents": 1,
        "level": "MEDIUM",
        "outcome": null,
        "status": "NEW",
        "title": "No title available",
        "url": "http://localhost/api/v1/alerts/4/"
    },
```


Kuvio 25 Cyphonin hälytysten API

5 YETI

YETI käyttää apunaan vähemmän tunnettuja ohjelmistoja Cyphoniin verrattuna. YETI:n tärkeimmät 3 ohjelmaa ovat

- MongoDB. Tietokantasovellus
- Redis. Tietokantasovellus joka noudattaa pääasiassa key-value tietorakennemallia.
- Celery. Työtehtävien ajastus-ohjelmisto.

YETI on avoimen lähdekoodin kehitysprojekti. YETI:n kehitys tapahtuu github-sivustolla. YETI:n tarkoituksena on tarjota nopea, tehokas ja yhteensopiva ohjelmisto. (Chopitea 2017.)

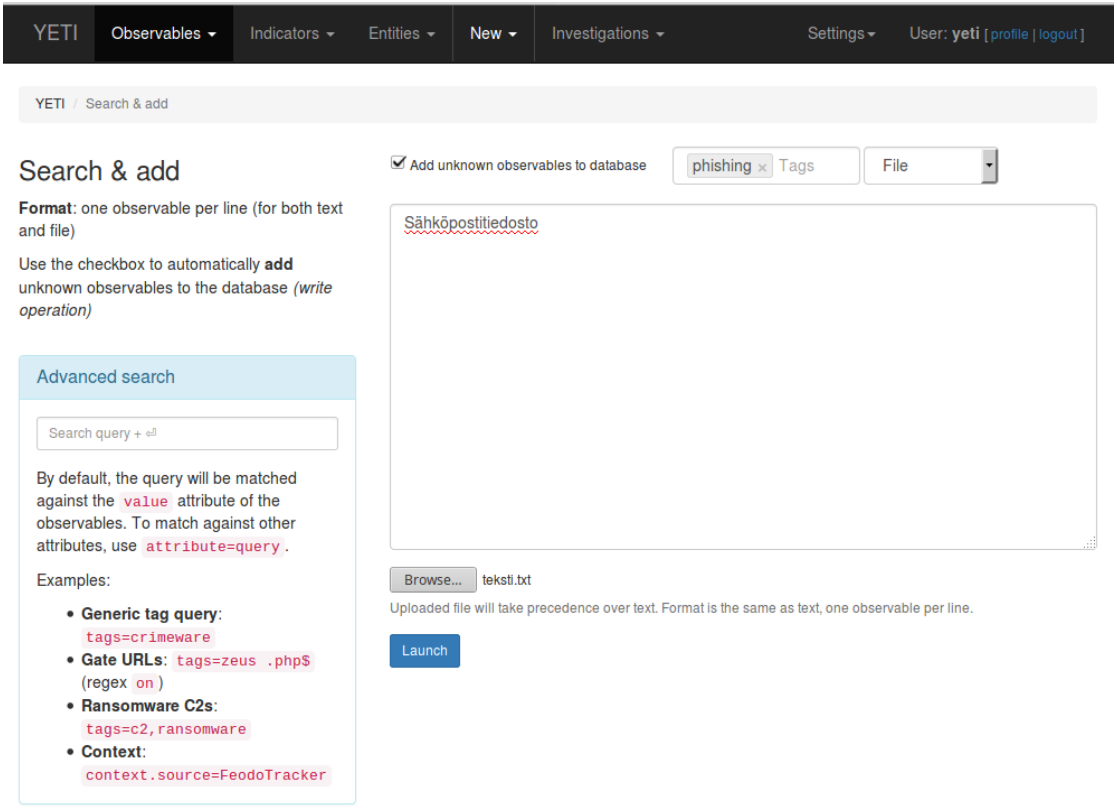
5.1 Asennus ja käyttöönotto

YETI on laajuudeltaan Cyphonia huomattavasti pienempi ohjelmisto. Sen vuoksi YETI on nopea asentaa. Myös käyttöönotto on yksinkertaista.

YETI:n asennus tapahtuu lataamalla käyttöjärjestelmään tarvittavat riippuvuudet, ja sen jälkeen lataamalla YETI:n ohjelmisto github:sta. Asennus tapahtuu komentoriviltä ja se tarvitsee vain muutaman komennon. Kun YETI:n web- palvelu on käynnistetty, web-käyttöliittymä löytyy osoitteesta: ”http://localhost:5000”.

5.2 Ominaisuudet ja käyttöliittymä

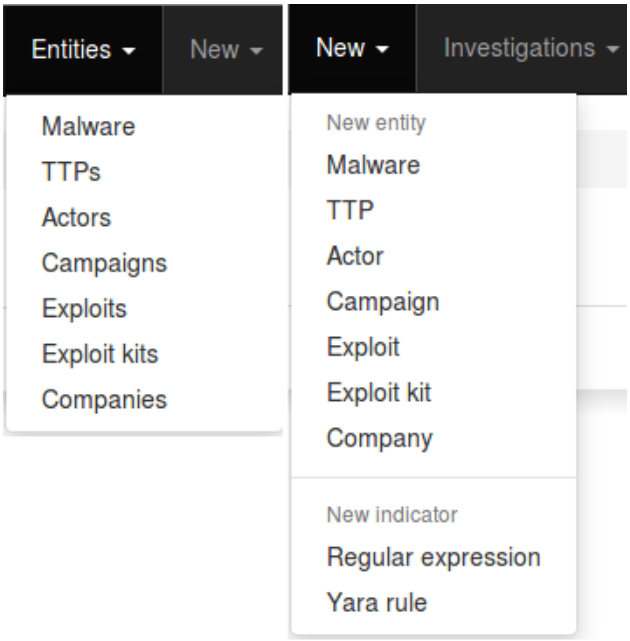
Observables-valikosta voidaan hakea ja lisätä tarkkailun alla olevia kohteita, sekä myös uploadata tiedostoja palvelimelle. Tämä sivu toimii myös YETI:n aloitussivuna. (Ks. Kuvio 26.)



Kuvio 26 YETI:n aloitussivusto

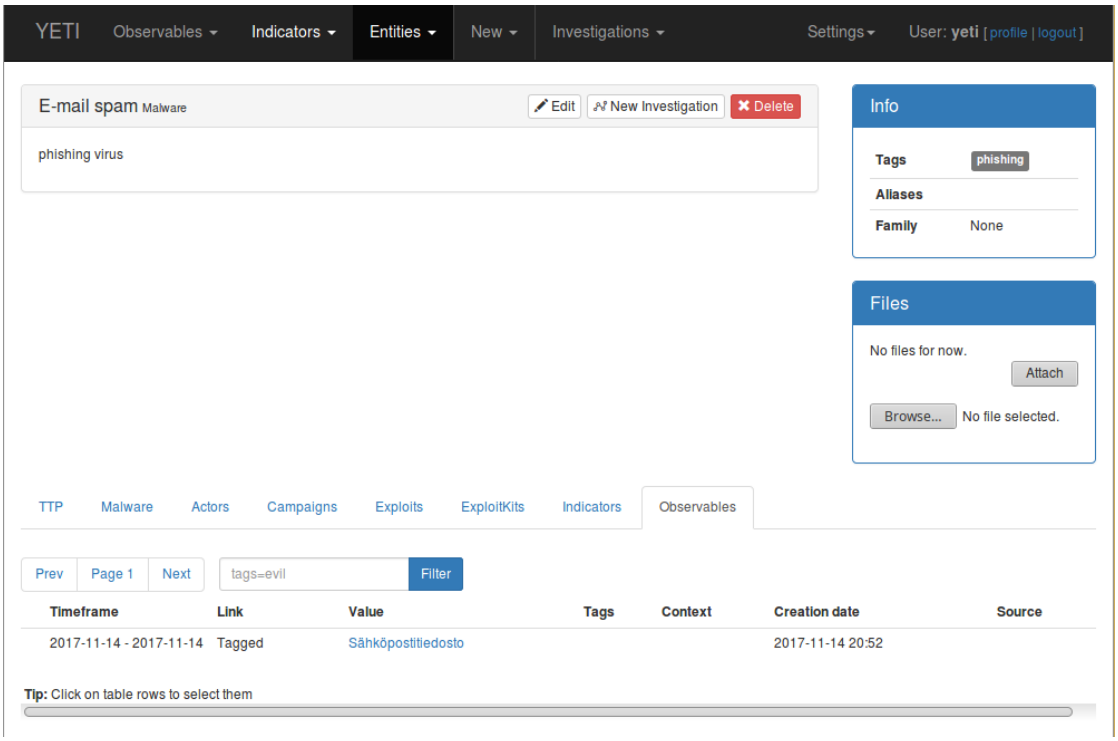
Indicators-välilehdeltä lisätään indikoivia tekijöitä, jotka ovat joko regex- tai YARA-muodossa. Indikaattoriksi voitaisiin esimerkiksi asettaa tiedostopolku ja kaikki sen alikansioissa sijaitsevat tiedostot. Jos tekijällä on monta eri tiedostoversiota, voidaan myös luoda regex-sääntö, jolla saadaan haettua kaikki kyseiset versiot.

Entities-välilehdeltä voidaan listata kaikki tietokannassa sijaitsevat eri tyypin kohteet. New-välilehdeltä voidaan lisätä uusia kohteita sekä myös indikoivia tekijöitä. (Ks. Kuvio 27.)



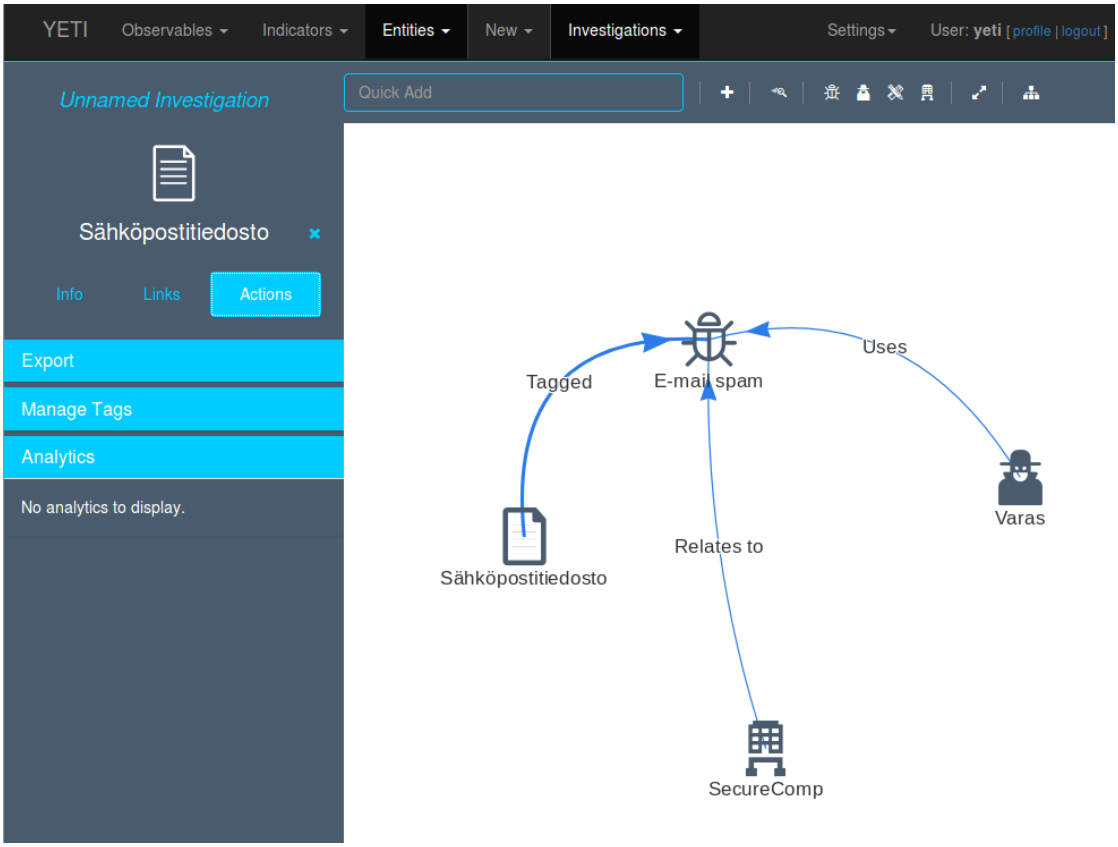
Kuvio 27 Entities- ja New-pudotusvalikot

Entity-tekijöille voidaan aloittaa tutkinta. Tutkintoja pääsee selaamaan Investigations-välilehdeltä. Tekijöille voidaan myös asettaa tarkkailtava kohde. Kohteeseen voidaan liittää lisätietoa kuten esim. kohdeyritys, TTP tai malware. Kohteelle voidaan myös uploadata tiedosto. Kuviossa 28 on YETI:n näkymä listattavasta kohteesta.



Kuvio 28 Kohteen listaus

Tutkintapaneeli tarjoaa graafisen käyttöliittymän tapauksia varten. Paneelin kautta voidaan lisätä kohdeyrityksiä, henkilöitä sekä liittää esim. malware-tiedosto havainnoitavaan tekijään. (Ks. Kuvio 29.)



Kuvio 29 Graafinen käyttöliittymä

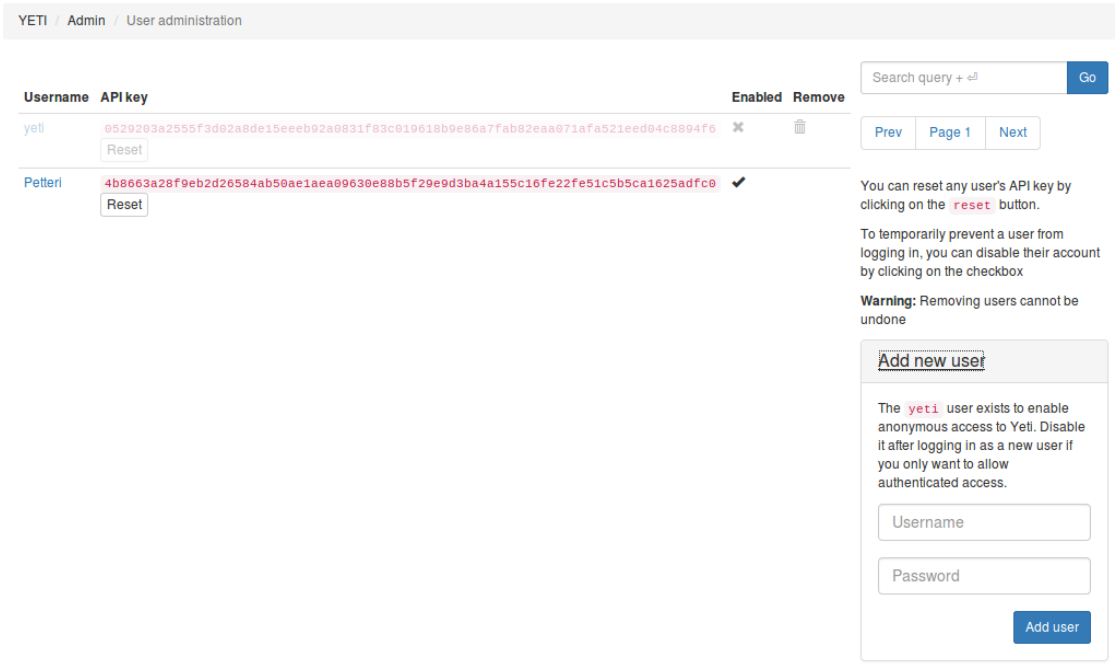
Settings-välilehdeltä voidaan hallinnoita käyttäjiä sekä YETI-järjestelmää. Samasta valikosta löytyy myös dataflows, analytics ja tags-välilehdet.

Analytics tarjoaa lisätoimintoja tietokannassa olevalle datalle, kuten esim. kohdeosoitteiden nimipalvelinhaun.

Dataflows-välilehdeltä hallinnoidaan feeds-toimintoa. Feeds-toiminnon avulla automatisoidaan tiedonkeräämisen prosesseja. Tags-välilehdellä voidaan hallinnoida tunnisteita ja esim. liittää kaksi tunnistetta yhteen.

Kuviossa 30 on esitettynä YETI:n käyttäjienhallinta.

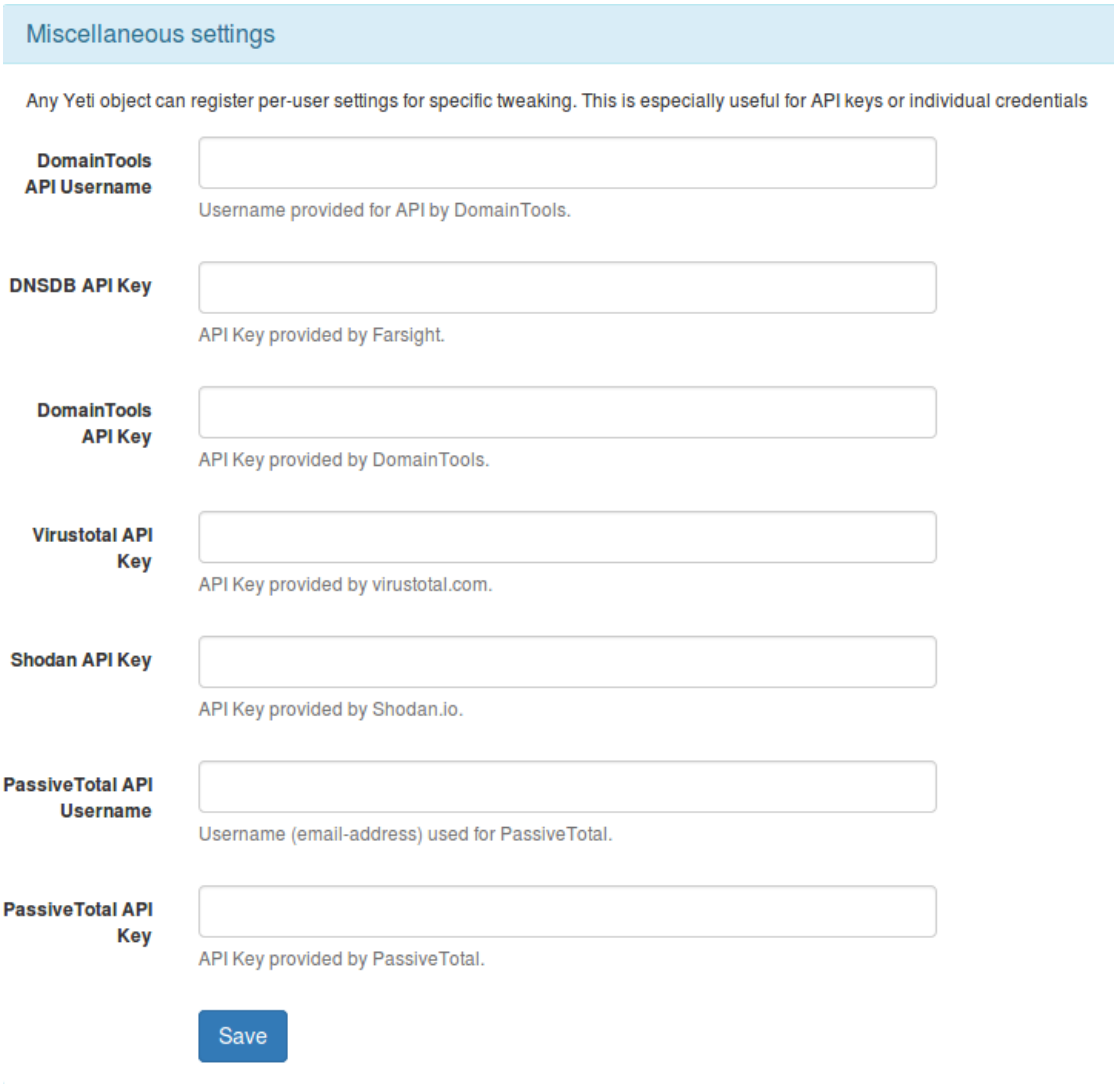
Oletuskäyttäjä tulee ottaa pois käytöstä ja luoda uusi käyttäjä. Huomattavaa on, että kaikki käyttäjät ovat samalla tasolla. Varsinaista admin-käyttäjää ei ole.



Kuvio 30 Käyttäjienhallinta

YETI voidaan integroida moniin muihin internetissä sijaitseviin palveluihin. Palvelut tarvitsevat esim. API-avaimen, joka voidaan lisätä YETI:in käyttäjäkohtaisesti käyttäjienhallinnasta. (Ks. Kuvio 31.)

Domaintools tekee nimipalvelinhaun. DNSDB on tietokanta-palvelu DNS-osoitteista. Virustotal-palvelun avulla käyttäjä voi ladata tiedoston sivustolle ja tarkistaa onko siinä viruksia. Shodan on hakupalvelu, jolla voidaan hakea verkossa sijaitsevia laitteita, kuten tietokoneita, reitittimiä ja palvelimia. Passivetotal-palvelun avulla voidaan tehdä nimipalvelinhakuja, WHOIS-hakuja sekä hakea tietoa TLS-sertifikaateista.



Kuvio 31 YETI:n lisätoiminnot

5.3 API

YETI:n API:n avulla voidaan hakea tarkkailtavia kohteita (observables), indikaattoreita sekä tekijöitä eli entities. YETI:stä löytyy myös API-toiminnallisuus ”Feeds and Exports” -toiminnallisuudelle. Molemmat edellä mainituista pitää luoda manuaalisesti, tai käyttää olemassa olevia malleja.

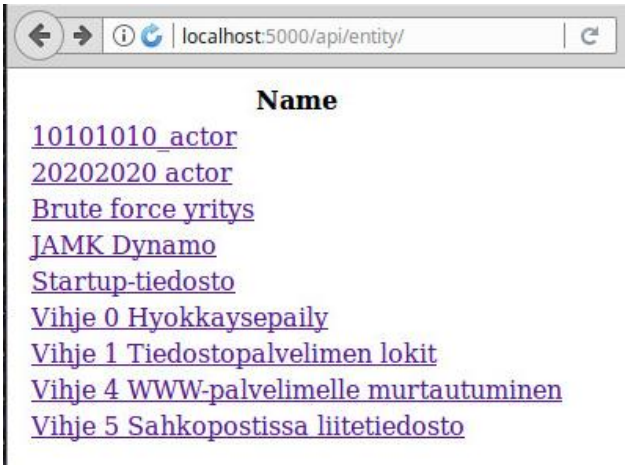
Työssä keskitytään GET-metodilla haettaviin perustietoihin. Taulukossa 7 on listattuna työn kannalta tärkeimmät API-toiminnot.

Taulukko 7. YETI:n API-toiminnallisuus

Sijainti	**Metodit**	**Selitys**
/api/observable/	GET /api/observable/ GET /api/observable/(id) POST /api/observable/	Hakee kaikki tarkkailtavat kohteet, tai yksittäisen ID:n perusteella. POST-komennolla luodaan uusi tekijä.
/api/tag/	DELETE /api/tag/(id) GET /api/tag/(id) GET /api/tag/ POST /api/tag/ POST /api/tag/(id)	Hakee, poistaa tai lisää tageja.
/api/indicator/	DELETE /api/indicator/(id) GET /api/indicator/(id) GET /api/indicator/ POST /api/indicator/ POST /api/indicator/(id)	Hakee, poistaa tai lisää indikaattoreita.
/api/entity/	GET /api/entity/ GET /api/entity/(id) DELETE /api/entity/(id) POST /api/entity/ POST /api/entity/(id)	Hakee kaikki tietokannassa olevat tekijät, tai vain yksittäisen kohteen ID:n perusteella. Voidaan luoda ja poistaa tekijöitä. Tekijä viittaa kaikkiin muuttujatekijöihin, eli malware, company ym.
/api/investigation/	DELETE /api/investigation/(id) GET /api/investigation/(id) GET /api/investigation/ POST /api/investigation/ POST /api/investigation/(id)	Haetaan, poistetaan tai lisätään tutkintoja.
Haku	POST /api/observablesearch/ POST /api/indicatorsearch/ POST /api/entitysearch/	Suorittaa haun POST-komennon avulla.

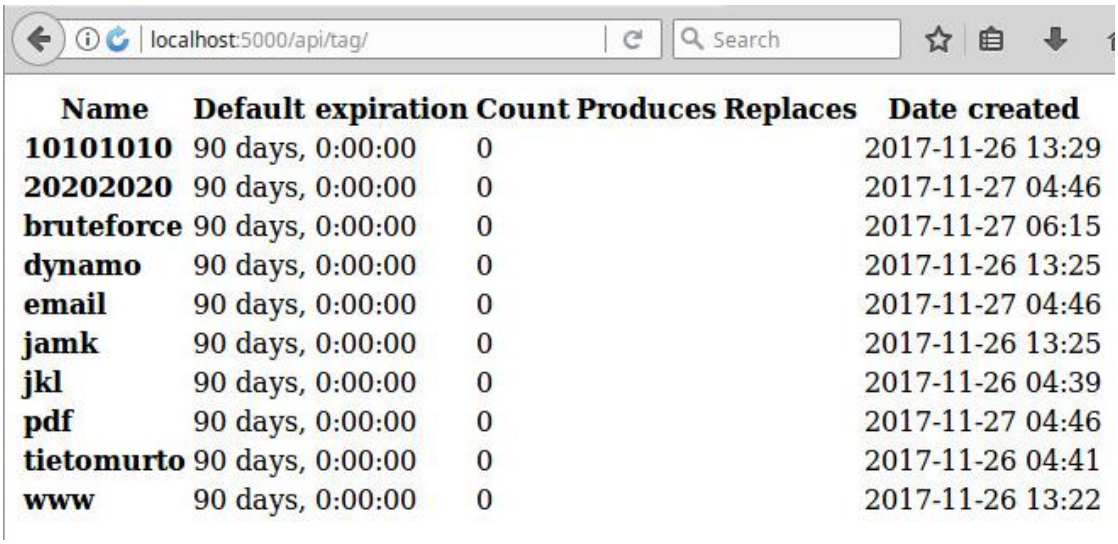
Kuviossa 32 on esitettynä kaikkien tekijöiden listaaminen.

YETI ei erottele tekijöitä niiden tyypin perusteella. Tämä huomataan siitä, että kuviossa 32 on listattuna sekä malware-, company-, actor- että campaign-tekijät.



Kuvio 32 Entities-tekijöiden listaus

Kuviossa 33 on listattuna kaikki tietokannassa olevat tagit.



Name	Default expiration	Count	Produces	Replaces	Date created
10101010	90 days, 0:00:00	0			2017-11-26 13:29
20202020	90 days, 0:00:00	0			2017-11-27 04:46
bruteforce	90 days, 0:00:00	0			2017-11-27 06:15
dynamo	90 days, 0:00:00	0			2017-11-26 13:25
email	90 days, 0:00:00	0			2017-11-27 04:46
jamk	90 days, 0:00:00	0			2017-11-26 13:25
jkl	90 days, 0:00:00	0			2017-11-26 04:39
pdf	90 days, 0:00:00	0			2017-11-27 04:46
tietomurto	90 days, 0:00:00	0			2017-11-26 04:41
www	90 days, 0:00:00	0			2017-11-26 13:22

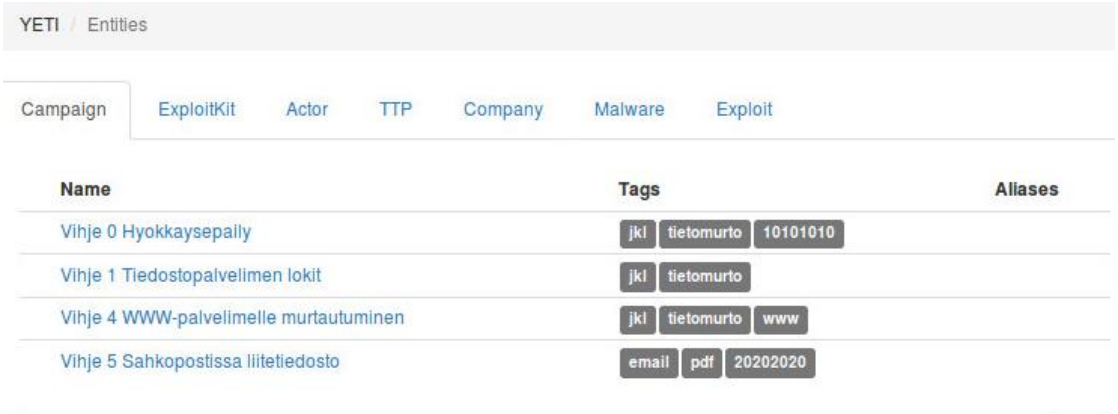
Kuvio 33 Tag-merkintöjen listaus

Investigations-API:n testaaminen antoi HTTP 500-virheilmoituksen. Pyyntö-otsikossa tulisi määrittää, että otamme vastaan application/json-tietotyyppiä, tai käyttää selaimen sijasta API-skriptiä jolla tietoja haetaan.

5.4 Poikkeamienhallinnan testi YETI:llä

Sovimme toimeksiantajan kanssa esimerkkitapauksen eli keissin, jonka avulla suorittaisimme lopullisen vertailun ohjelmistojen välillä. Koska Cyphon ei tarjonnut tähän soveltuvaa toiminnallisuutta, testi ajettiin vain YETI:llä.

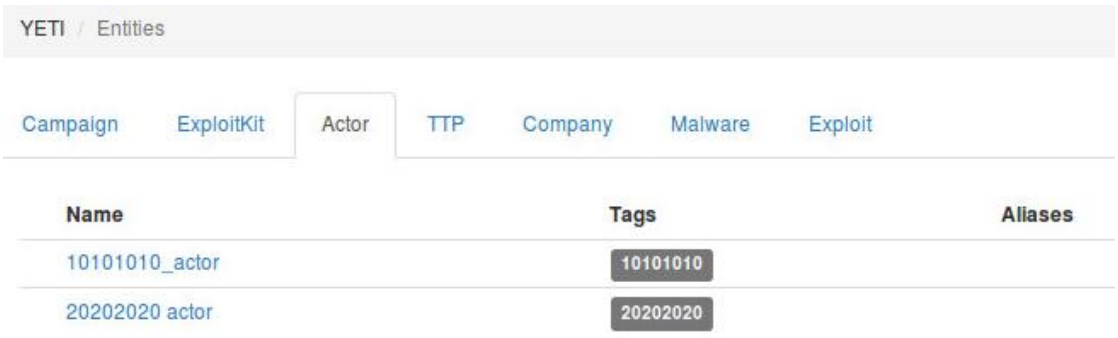
Keissin suorittaminen alkoi kirjaamalla tapahtuneet poikkeamat ylös. Ensimmäisenä oli pelkkä epäily tietomurrosta. Sen jälkeen kirjattiin 3 muuta tapausta. (Ks. Kuvio 34.)



Kuvio 34 Kirjatut poikkeamatapaukset

Huomattavaa on, että erikoismerkkien tai skandinaavisten kirjaimien käyttäminen ei toiminut ohjelmistossa kunnolla. Poikkeaman otsikoksi sai laitettua ”#0 Hyökkäysepäily”, mutta tällöin tagin lisääminen ei toiminut. Ongelma ratkesi poistamalla skandinaaviset kirjaimet sekä erikoismerkit. Alaviivan tai normaalin viivan käyttäminen eivät tuottaneet ongelmaa.

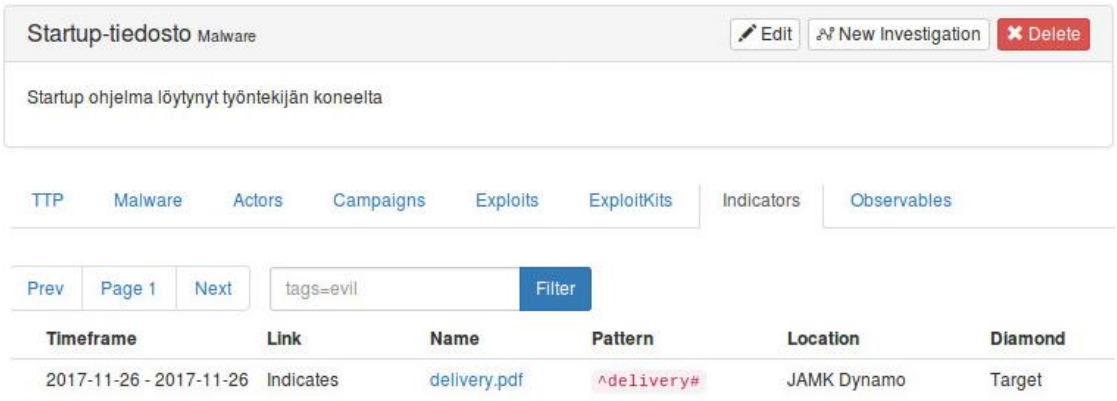
Myöhemmin keississä ilmeni palvelimen lokeista kaksi eri sijainneissa olevaa IP-osoitetta, jotka olivat 10.10.10.10 sekä 20.20.20.20. IP-osoitteet kirjattiin ylös ”Actor”-, eli hyökkääjä-välilehdelle. Pisteiden käyttäminen IP-osoitteessa johti tagin lisäämisen epäonnistumiseen. (Ks. Kuvio 35.)



Kuvio 35 Hyökkääjien lisääminen

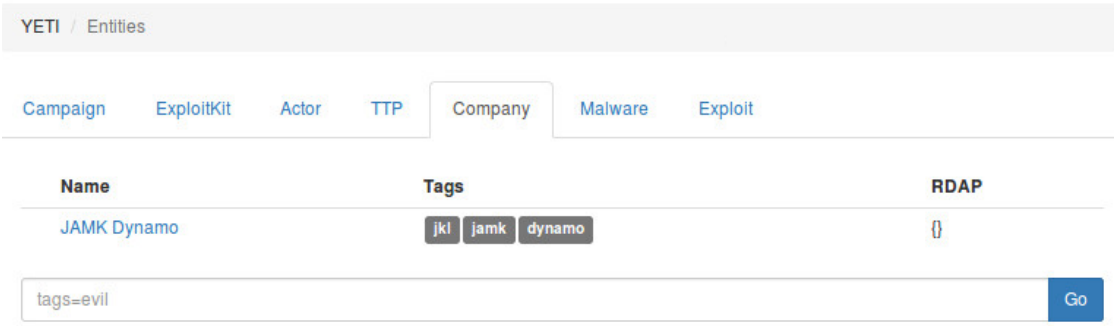
Seuraavana ohjelmaan lisättiin merkintä uudesta haittaohjelmasta, mutta mitään varsinaista tiedostoa ei lisätty ohjelmistoon. Tiedosto oli muodossa ”delivery.pdf”, ja

sitä varten luotiin regex-indikaattori. Indikaattori ja malware-kohde linkitettiin toisiinsa. (Ks. Kuvio 36.)



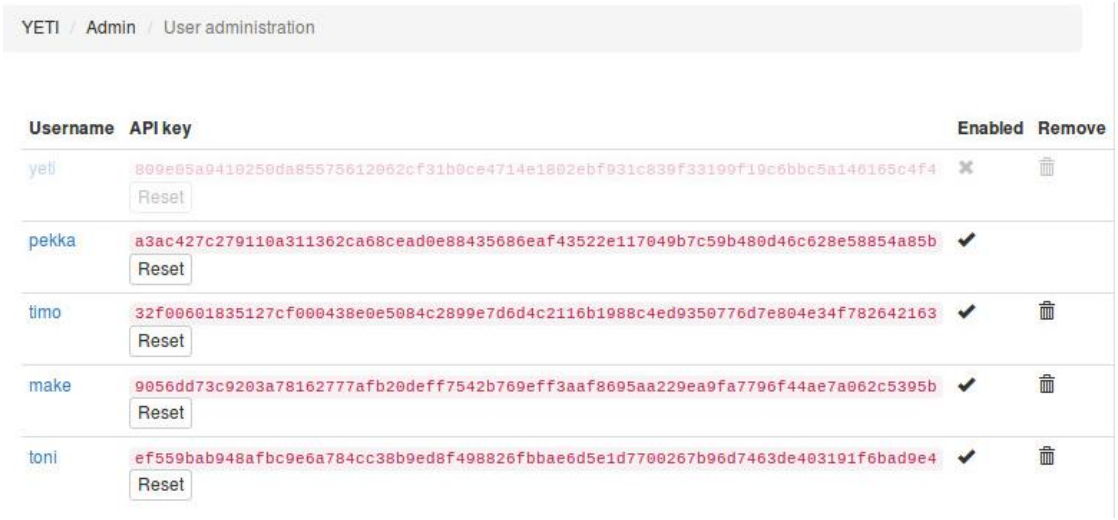
Kuvio 36 Malware-kohteen lisääminen

Ohjelmistoon lisättiin myös esimerkkinä toimiva kohdeyritys ja sille lisättiin tageja. (Ks. Kuvio 37.)



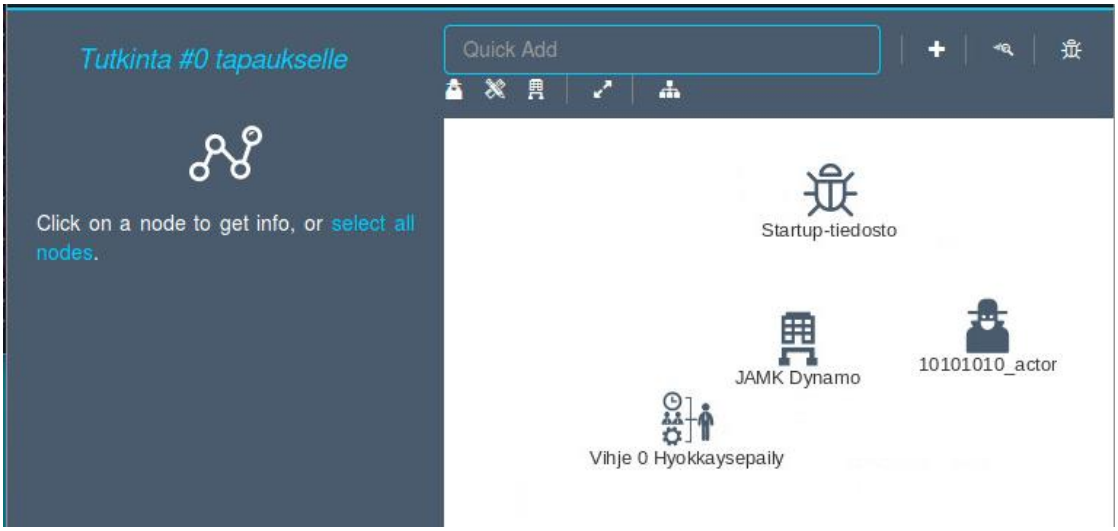
Kuvio 37 Yrityksen lisääminen

Keissiä varten ohjelmistoon lisättiin neljä uutta käyttäjää. Kaikki käyttäjät ovat samalla tasolla eikä YETI:stä löytynyt mitään toiminnallisuutta käyttäjien hallitsemiselle, eikä myöskään käyttäjien tai vastuuhenkilöiden lisäämiseen tutkintoihin. (Ks. Kuvio 38.)



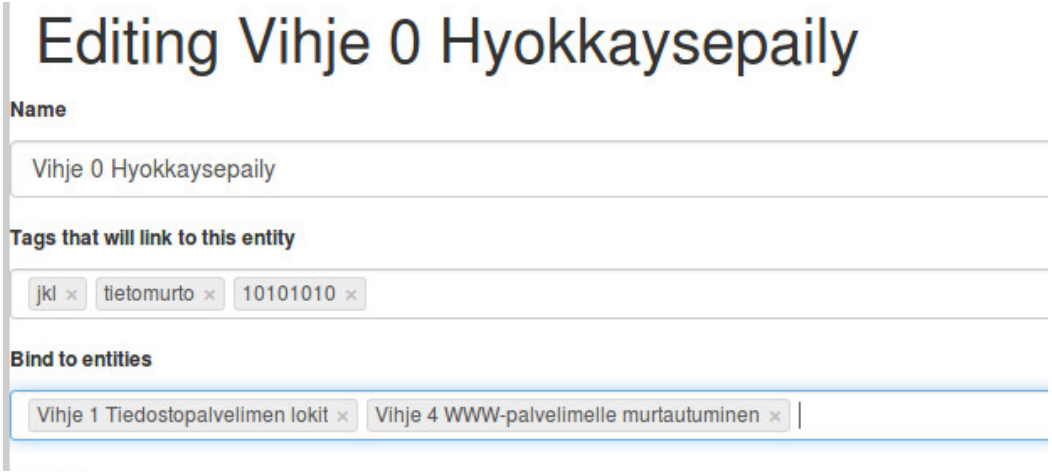
Kuvio 38 Käyttäjien listaaminen

Ensimmäiselle vihjeelle luotiin uusi tutkinta graafisessa käyttöliittymässä. Muuttujina käytettiin selville saatua tietoa. Hyökkäys tuli 10.10.10.10-osoitteesta käyttäen apunaan startup-tiedostoa. (Ks. Kuvio 39.)



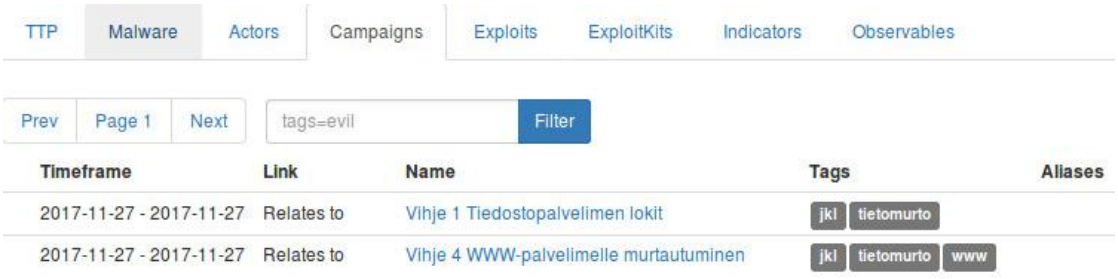
Kuvio 39 Tutkinnan lisääminen

Myöhemmin tapauksessa huomataan, että samasta 10.10.10.10 IP-osoitteesta on tehty kaksi eri hyökkäystä. Tapaukset linkitetään bind-toiminnolla toisiinsa. (Ks. Kuvio 40.)



Kuvio 40 Tapausten liittäminen toisiinsa

Tämän jälkeen #0-kampanjassa näkyy linkitettyinä kaksi muuta kampanja-tapausta. (Ks. Kuvio 41.)



Kuvio 41 Liitetyt tapaukset

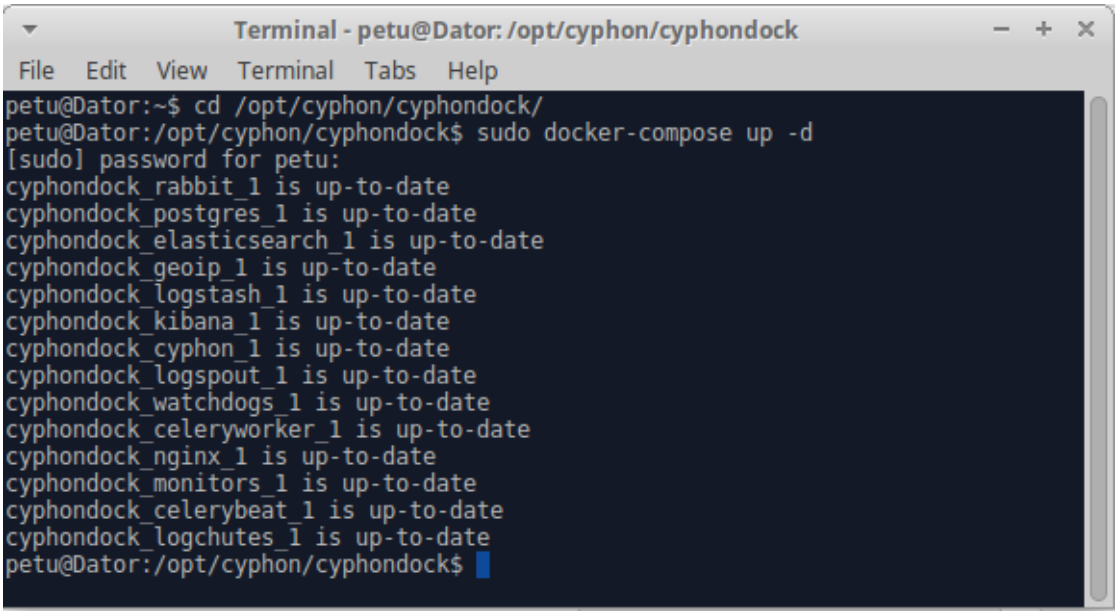
Toimeksiantajan hakemaa aikajana-toiminnallisuutta YETI-ohjelmistosta ei löydy. Tapauksia ei ole myöskään mahdollista saattaa loppuun tai asettaa tapauksille vastuuhenkilöitä. Ohjelmiston toiminnallisuus on näiden muuttujien pohjalta puutteellista. Myös YETI:n API on erittäin yksinkertainen tekstipohjainen versio. Cyphon tarjoaa kattavamman API-toiminnallisuuden. Myös kaksi edellä mainittua käyttäjienhallinnan toiminnallisuutta löytyvät. (Ks. Kuvio 42.)

Kuvio 42 Tapausten antaminen käyttäjille Cyphonissa

6 Vertailu

YETI on hyvä valinta pienelle yritykselle koska se on helppo ja nopea asentaa. Käyttöliittymä ja toiminnallisuus ovat myös selkeitä. Cyphonin asentaminen oli epäselvempää. Asennus kesti kauemmin sekä lataamis- että asennusvaiheessa.

Cyphon käynnistetään ja sammutetaan komentoriviltä. Komentorivin ulkoasu ei ole tarpeeksi selkeä, jotta käyttäjä tietäisi, että koska ohjelma on lopulta käynnistynyt. Ohjelmat siis käynnistyvät taustalla pidemmän aikaa. (Ks. Kuvio 43.)
Nginx antoi HTTP 502-virheilmoituksen, jos Cyphon ei ollut ehtinyt käynnistyä tarpeeksi pitkälle. Tämä lisäsi satunnaisuuden tunnetta Cyphonin käytössä.



Kuvio 43 Cyphon käynnistys komentoriviltä

Cyphon vaati myös monta askelta tiedon säilömistä varten. Piti luoda pullo, sille leima, säiliö ja sitten luoda monivaiheinen tiedonkäsittelyn prosessi. Monipuolisesta tiedonkäsittelystä on hyötyä ohjelman laajemmassa käytössä, mutta prosessi on haastava ensikäyttäjälle.

Cyphon vaatii myös enemmän tehoja järjestelmältä. Minimivaatimuksena on 8GB muistia, tuplaydin-prosessori sekä 20GB kovalevytilaa.

Plussaa Cyphon saa paremmasta käyttäjien- ja ryhmien-hallinnasta. YETI:ssä jokainen käyttäjä oli samalla tasolla.

YETI-ohjelmistoa käyttäessä ei tullut montaa virheilmoitusta. Muutama bugi ilmeni, jotka liittyivät skandinaavisiin kirjaimiin ja erikoismerkkeihin.

YETI:n huonona puolena ovat ohjelmiston rajallisuus ja toiminnallisuuden puute.

Jos ohjelmisto tulisi ottaa nopeasti käyttöön, tällöin Cyphon olisi liian työläs. Tiedonkäsittelyn prosessiin sekä tiedon säilömiseen tulee perehtyä pitkään. Tämän pohjalta pitää luoda suunnitelma. Käyttäjän pitää olla myös hyvin selvillä siitä, miten tiedonkäsittelyn prosessi toimii. Mielestäni tässä vaiheessa oli liian monta muuttujaa, ellei ohjelmistoon halua perehtyä 100-prosenttisesti.

Cyphon ei siis ole ns. valmis paketti, vaan se on konfiguroitava täysin itse. YETI taas on heti käyttövalmis testausta varten.

7 Pohdinta

Olen erittäin tyytyväinen aihealueeseen. Incident Response ei ollut terminä millään tapaa minulle tuttu vaikka olinkin suorittanut tietoturvan opintoja. En myöskään ollut koskaan pohtinut tarkemmin miten API:t toimivat tai miten web-palveluiden sovelluskehitys tapahtuu. Työ antoi loistavan mahdollisuuden tutustua käytännössä API:en toimintaan. Iso osa teoriasta oli uutta materiaalia, joten uutta tuli opittua paljon sekä tehostettua vanhaa tietoa.

Työtä varten löytyi hyvin lähteitä. Päälähteenä teorialle käytettiin books24x7-verkkopalvelua. Tärkeänä lähteenä toimi mm. vuonna 2017 kirjoitettu kirja API-teknologioista. Oli hyvä, että tarjolla oli ajantasaista tietoa tuoreista teknologioista. Osa teoriasta oli taas ns. historiallisempaa materiaalia ja näihin materiaali löytyi 90-luvun teoksista.

Työn määrittely sujui valitettavasti huonosti. Emme saaneet toimeksiantajan kanssa kommunikoitua tarpeeksi selväksi työn lopullista merkitystä. Työn tekeminen helpottui kun oli selvillä se, että tarkalleen ottaen mitä näistä ohjelmistoista piti vertailla ja että mihin tarkoitukseen nämä ohjelmistot voisivat tulla käyttöön.

Se, että Cyphonista ei löytynyt toiminnallisuutta tapausten lisäämiseen manuaalisesti haittasi opinnäytetyön lopullista tavoitetta. Lopullinen tutkimuskysymys eli ohjelmistojen soveltuvuus poikkeamienhallintaan tuli selvitettyä hyvin. YETI sopii manuaaliseen tapausten lisäämiseen ja Cyphonista löytyy toiminnallisuuksia joiden avulla voidaan luoda hälytyksiä automaattisesti. Työ antaa toimeksiantajalle kattavan kuvauksen siitä, että millaiset ohjelmat ovat kyseessä ja millainen käyttöliittymä ja toiminnallisuus niissä on.

Valitettavasti avoimen lähdekoodin IR-ohjelmat ovat vasta pitkälti kehitysvaiheessa. Tutkittavaa olisi ollut varmasti enemmän, jos ohjelmistot olisivat olleet pidemmälle kehitettyjä. Olisin toivonut että YETI:stä olisi löytynyt kattavampi API-toiminnallisuus. Se olisi myös helpottanut työtä, jos Cyphon olisi tarjonnut manuaalisen lisäystoiminnon hälytyksille. Uskon että tulevaisuudessa tämä toiminnallisuus tulee löytymään, koska nykyisen automatisoitun mallin käyttöönottokynnys on korkea. Jos tuote on kokonaan automatisoitu, niin tällöin tarvitaan rinnalle toinen tuote johon

kirjataan manuaaliset hälytykset muista lähteistä. Cyphonille on tärkeätä, että tietorakenne ei mene rikki tai muutu liikaa. Manuaalinen lisäystoiminto voitaisiin toteuttaa PostgreSQL-tietokannan avulla. Automaattisesti kerätty tieto lisättiin siis Elasticsearch- tai MongoDB-tietokantoihin.

Cyphon kaipaisi myös parempaa ja kattavampaa dokumentointia. Osa ilmenneistä bugeista oli helposti ratkaistavissa sovelluskehittäjän näkökulmasta, mutta ohjeissa niistä ei tullut mitään mainintaa. Manuaalisen asentamisen eli ilman dockeria asentamisen ohjeet olivat myös todella heikot. Iso osa tarvittavista riippuvuuksista oltiin jätetty listaamatta, joten näitä puuttuvia lisäosia piti hakea Googlella virheilmoitusten perusteella. Dokumentoinnin puute hidasti myös työn etenemistä ja alkuun pääsyä. Olisin kaivannut suoraviivaisemmat ohjeet.

Jatkotutkimuksena lähtisin tutkimaan ohjelmistojen lisätoiminnallisuuksia jotka jäivät opinnäytetyön aihealueen ulkopuolelle. Näitä ovat mm. YETI:n Feeds & Exports -toiminto sekä Cyphonin käyttäminen lokien analysointiin. Lokien tutkimista varten tulisi kehittää uudet tietorakenteet, hakuparametrit sekä suorittaa konfigurointi kuntoon.

Työn tutkimuskysymyksenä oli näiden ohjelmistojen soveltuvuus IR-toimintaan, johon tuli vastattua hyvin. Cyphonin tarkempi konfiguroiminen alkoi kuitenkin siirtyä pois alkuperäisestä tutkimuskysymyksestä. Siitä saisi kirjoitettua kokonaisen erillisen työn.

Teorian kirjoittaminen sujui erittäin hyvin. Ohjelmistojen asennus ja käyttöönotto olisivat voineet tapahtua omalta kohdaltani nopeammin, joka olisi nopeuttanut työssä loppuun pääsyä. Olen lopputulokseen ja kirjoitusrakenteeseen tyytyväinen.

Lähteet

Amoroso, E. 2011. Cyber Attacks: Protecting National Infrastructure. Burlington: Elsevier

Brajesh, D. 2017. API Management: An architect's guide to developing and managing API's for your organization. Intia:Apress

Chopitea, T. 2017. Introducing yeti. Verkkomateriaali. Viitattu 1.12.2017. https://yeti-platform.github.io/introducing-yeti

Cichonski, P., Millar, T., Grance, T. & Scarfone, K. 2012 Computer Security Incident Handling guide. Viitattu 24.10.2017 http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf

Connolly, D. 2003. Development History. Verkkomateriaali. Viitattu 1.12.2017. https://www.w3.org/XML/hist2002

Data processing. 2017. Dunbar Security Solutions Inc. Tekninen manuaali. Viitattu 21.11.2017. http://cyphon.readthedocs.io/en/latest/user-manual/data-processing.html

Data storage. 2017. Dunbar Security Solutions Inc. Tekninen manuaali. Viitattu 24.11.2017. http://cyphon.readthedocs.io/en/latest/user-manual/data-storage.html

Endsley, M. 1995. Toward a theory of situation awareness in dynamic systems. Tiedejulkaisu. Viitattu 28.10.2017 https://www.researchgate.net/profile/Mica_Endsley/publication/210198492_Endsley_MR_Toward_a_Theory_of_Situation_Awareness_in_Dynamic_Systems_Human_Factors_Journal_371_32-64/links/548f61bf0cf214269f263b08/Endsley-MR-Toward-a-Theory-of-Situation-Awareness-in-Dynamic-Systems-Human-Factors-Journal-371-32-64.pdf

Fielding, R. 2000, Representational State Transfer (REST), Viitattu 29.10.2017 https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Gaining the advantage. Applying Cyber Kill Chain methodology to network defense. Lockheed Martin. 2015. Viitattu 20.11.2017. https://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/Gaining_the_Advantage_Cyber_Kill_Chain.pdf

Gibson, D. 2015. Managing Risk in Information Systems. 2. painos. Massachusetts: Jones & Bartlett Learning

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., Nielsen, H., Karmarkar, A. & Lafon, Y. 2007, SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). Verkkojulkaisu. Viitattu 29.10.2017 https://www.w3.org/TR/soap12/

Heimonen, J. 2017. Avoimen lähdekoodin työkalujen tutkiminen ja vertailu tilannetietoisuutta varten poikkeamanhallinnassa. Opinnäytetyö. Viitattu 1.12.2017. http://www.theseus.fi/bitstream/handle/10024/125592/Heimonen_Jere.pdf

Johnson, L. 2013. Computer Indicent Response and Forensics Team Management Conducting a Successful Incident Response. Massachutess: Elsevier Science

JSON Syntax. N.d. Opetusmateriaali. w3schools. Viitattu 1.11.2017 https://www.w3schools.com/js/js_json_syntax.asp

JSON vs XML. N.d. Opetusmateriaali. w3schools. Viitattu 1.11.2017 https://www.w3schools.com/js/js_json_xml.asp

Kim, D., Solomon, M. 2014. Fundamentals of Information System Security. 2. painos. Massachusetts: Jones and Bartlett Learning

Overview – Cyphon 1.5.3 documentation. 2017. Dunbar Security Solutions Inc.. Viitattu 3.11.2017. http://cyphon.readthedocs.io/en/latest/overview.html

Raggad, B. 2010, Information Security Management: Concepts and Practice. Flora:Auerbach Publications

RFC 2616. 1999. Hypertext Transfer Protocol – HTTP/1.1. Viitattu 29.10.2017. https://tools.ietf.org/html/rfc2616

RFC 3986. 2005. Uniform Resource Identifier (URI): Generic Syntax. Viitattu 1.11.2017. https://www.ietf.org/rfc/rfc3986.txt

Räsänen, H. N.d. Kvalitatiiviset tutkimusmenetelmät. Opetusmateriaali. Viitattu 1.12.2017. http://www.hamk.fi/verkostot/kudos/menetelmat/Documents/4_Kvalitatiiviset_tutkimusmenetelmaet.pdf

Schperberg, R & Brancik, C. 2005. Cybercrime – Incident Response and Digital forensics. Illinoid: ISACA

Taylor, J. 2012. Decision Management Systems: A Practical guide to using business rules and predictive analytics. Boston: IBM Press

The OODA Loop: A Holistic Approach to Cyber Security. TK Keanini. 2014. Viitattu 24.10.2017. https://www.lancope.com/blog/ooda-loop-holistic-approach-cyber-security

Tietoa meistä. N.d., JYVSECTEC. Viitattu 29.11.2017. https://jyvsectec.fi/fi/tietoa-meista/

XML tree. N.d. Opetusmateriaali. w3schools. Viitattu 1.11.2017 https://www.w3schools.com/xml/xml_tree.asp