Yapılandırma

Bu kılavuz, TheHive platformunun yapılandırma ayarlarını yönetmek için kullanılan yöntemleri ve araçları tanıtmaktadır.

- Veritabanı ve Dizin Yapılandırması
- Dosya Depolama Yapılandırması
- Bağlayıcılar Yapılandırması
- Akka Yapılandırması
- Logs (Günlükler) Yapılandırması
- Proxy Yapılandırması
- Secrets (Gizli) Yapılandırma
- SSL Yapılandırması
- Hizmet Yapılandırması
- Uyumluluk

Veritabanı ve Dizin Yapılandırması

TheHive, verileri ve dizini yönetmek için Cassandra ve Elasticsearch veritabanlarını kullanır. Kuruluma göre, örnek kullanabilir:

Temel Yapılandırma

TheHive için tipik bir veritabanı yapılandırması şu şekildedir:

```
## Database configuration
db {
  provider = janusgraph
  janusgraph {
    ## Storage configuration
    storage {
      backend = cql
      hostname = ["IP_ADDRESS"]
      cql {
        cluster-name = thp
        keyspace = thehive
      }
    }
    ## Index configuration
    index.search {
      backend = elasticsearch
      hostname = ["127.0.0.1"]
      index-name = thehive
    }
  }
}
```

Mümkün parametrelerin listesi

Parametre Tipi Açıklama

Parametre	Tipi	Açıklama
`provider`	string	provider adı. Varsayılan `janusgraph`
`storage`	dict	storage configuration (deepolama yapılandırması)
`storage.backend`	string	storage type. Can be `cql` or `berkeleyje` (depolama türü.)
`storage.hostname`	list of string	list of IP addresses or hostnames when using `cql` backend (cql arka ucunu kullanırken IP adreslerinin veya ana bilgisayar adlarının listesi)
`storage.directory`	string	local path for data when using `berkeleyje` backend (BerkeleyJE arka uç kullanıldığında veri için yerel yol.)
`storage.username`	string	account username with `cql` backend if Cassandra auth is configured (Eğer Cassandra kimlik doğrulaması yapılandırılmışsa, cql arka uç ile hesap kullanıcı adı.)
`storage.password`	string	account password with `cql` backend if Cassandra auth is configured (Cassandra kimlik doğrulaması yapılandırıldıysa, cql arka uç için hesap şifresi.)
`storage.port`	integer	port number with `cql` backend (`9042` by default). Change this if using an alternate port or a dedicated port number when using SSL with Cassandra (Cql arka uç ile bağlantı noktası numarası (varsayılan olarak 9042). Cassandra ile SSL kullanıyorsanız alternatif bir bağlantı noktası veya özel bir bağlantı noktası numarası kullanmak için bunu değiştirin.)
`storage.cql`	dict	configuration for `cql` backend *if used* (Kullanıldığında cql backend için yapılandırma.)

Parametre	Tipi	Açıklama
`storage.cql.cluster-name`	string	name of the cluster name used in the configuration of Apache Cassandra
`storage.cql.keyspace`	string	Keyspace name used to store TheHive data in Apache Cassandra
`storage.cql.ssl.enabled`	boolean	`false` by default. set it to `true` if SSL is used with Cassandra (Apache Cassandra yapılandırmasında kullanılan küme adı.)
`storage.cql.ssl.truststore.location`	string	path the the truststore. Specify it when using SSL with Cassandra (TheHive verilerini depolamak için Apache Cassandra'da kullanılan keyspace adı.)
`storage.cql.ssl.password`	string	password to access the truststore (Güven deposuna erişmek için şifre)
`storage.cql.ssl.client-authentication-enabled`	boolean	Enables use of a client key to authenticate with Cassandra (Cassandra ile kimlik doğrulaması için bir istemci anahtarı kullanımını etkinleştirir.)
`storage.cql.ssl.keystore.location`	string	path the the keystore. Specify it when using SSL and client auth. with Cassandra (Keystore'un yolu. Cassandra ile SSL ve istemci kimlik doğrulaması kullanılırken belirtiniz.)
`storage.cql.ssl.keystore.keypassword`	string	password to access the key in the keystore (Anahtara erişmek için şifre)
`storage.cql.ssl.truststore.storepassword`	string	password the access the keystore (Keystore'a erişim için şifre)
`index.search`	dict	configuration for indexes (Dizinler için yapılandırma)

Parametre	Tipi	Açıklama
`index.search.backend`	string	index engine. Default: `lucene` provided with TheHive. Can also be `elasticsearch` (Dizin motoru. Varsayılan: TheHive ile sağlanan lucene. Ayrıca elasticsearch olabilir.)
`index.search.directory`	string	path to folder where indexes should be stored, when using `lucene` engine (Lucene motoru kullanıldığında dizinlerin depolanacağı klasörün yolu.)
`index.search.hostname`	list of string	list of IP addresses or hostnames when using `elasticsearch` engine (Elasticsearch motoru kullanıldığında IP adresleri veya ana bilgisayar adlarının listesi.)
`index.search.index-name`	string	name of index, when using `elasticseach` engine (Elasticsearch motorunu kullandığınızda dizin adı.)
`index.search.elasticsearch.http.auth.type: basic`	string	`basic` is the only possible value (basic tek mümkün değerdir.)
`index.search.elasticsearch.http.auth.basic .username`	string	Username account on Elasticsearch (Elasticsearch'teki kullanıcı adı hesabı)
`index.search.elasticsearch.http.auth.basic .password`	string	Password of the account on Elasticsearch (Elasticsearch üzerindeki hesabın şifresi)
`index.search.elasticsearch.ssl.enabled`	boolean	Enable SSL `true/false` (SSL'yi etkinleştir true/false)
`index.search.elasticsearch.ssl.truststore.l ocation`	string	Location of the truststore (Güven deposunun konumu)

Parametre	Tipi	Açıklama
`index.search.elasticsearch.ssl.truststore.password`	string	Password of the truststore (Güven deposunun şifresi)
`index.search.elasticsearch.ssl.keystore.location`	string	Location of the keystore for client authentication (İstemci kimlik doğrulaması için keystore'un konumu)
`index.search.elasticsearch.ssl.keystore.storepassword`	string	Password of the keystore (Keystore'un şifresi)
`index.search.elasticsearch.ssl.keystore.keypassword`	string	Password of the client certificate (İstemci sertifikasının şifresi)
`index.search.elasticsearch.ssl.disable-hostname-verification`	boolean	Disable SSL verification `true/false` (SSL doğrulamasını devre dışı bırak true/false)
`index.search.elasticsearch.ssl.allow-self-signed-certificates`	boolean	Allow self signe certificates `true/false` (Kendinden imzalı sertifikalara izin ver true/false)

Not: İlk başlatma veya dizinleri yapılandırdıktan sonraki ilk başlatma, veritabanı büyük miktarda veri içeriyorsa biraz zaman alabilir. Bu süre, dizinlerin oluşturulmasından kaynaklanır.

Kullanım durumları

Veritabanı ve dizin motoru, kullanım durumuna ve hedef kuruluma bağlı olarak farklı olabilir:

Cassandra & Elasticsearch ile bağımsız sunucu :

1. Install a Cassandra server locally
2. Install Elasticsearch
3. Configure TheHive accordingly

```
## Database Configuration
db {
  provider = janusgraph
  janusgraph {
```

```
    ## Storage configuration
    storage {
      backend = cql
      hostname = ["127.0.0.1"]
      ## Cassandra authentication (if configured)
      username = "thehive_account"
      password = "cassandra_password"
      cql {
        cluster-name = thp
        keyspace = thehive
      }
    }
    ## Index configuration
    index.search {
      backend = elasticsearch
      hostname = ["127.0.0.1"]
      index-name = thehive
    }
  }
```

```
## Database Configuration
db {
  provider = janusgraph
  janusgraph {
    ## Storage configuration
    storage {
      backend = cql
      hostname = ["127.0.0.1"]
      ## Cassandra authentication (if configured)
      username = "thehive_account"
      password = "cassandra_password"
      cql {
        cluster-name = thp
        keyspace = thehive
      }
    }
    ## Index configuration
    index.search {
      backend = elasticsearch
      hostname = ["127.0.0.1"]
```

```
    index-name = thehive
  }
}
```

Cassandra ve Elasticsearch ile Küme:

1. Install a cluster of Cassandra servers
2. Get access to an Elasticsearch server
3. Configure TheHive accordingly

```
## Database Configuration
db {
 provider = janusgraph
 janusgraph {
  ## Storage configuration
  storage {
   backend = cql
   hostname = ["10.1.2.1", "10.1.2.2", "10.1.2.3"]
   ## Cassandra authentication (if configured)
   username = "thehive_account"
   password = "cassandra_password"
   cql {
    cluster-name = thp
    keyspace = thehive
   }
  }
  ## Index configuration
  index {
   search {
    backend  = elasticsearch
    hostname  = ["10.1.2.5"]
    index-name  = thehive
    elasticsearch {
     http {
      auth {
       type = basic
       basic {
        username = httpuser
        password = httppassword
       }
```

```
        }
      }
      ssl {
        enabled = true
        truststore {
          location = /path/to/your/truststore.jks
          password = truststorepwd
        }
      }
    }
  }
 }
}
```

Bu yapılandırmada, tüm TheHive simgerleri aynı yapılandırmaya sahip olmalıdır. Elasticsearch yapılandırması script.allowed_types için varsayılan değeri kullanmalı veya aşağıdaki yapılandırma satırını içermelidir.

```
script.allowed_types: inline,stored
```

Dosya Depolama Yapılandırması

Dosya depolama yapılandırması#

TheHive yerel veya dağıtılmış dosya sistemlerini kullanacak şekilde yapılandırılabilir.

Local veya NFS :

Özel klasör oluşturun; thehive:thehive kullanıcı ve grubuna ait olmalıdır.

```
mkdir /opt/thp/thehive/files
chown thehive:thehive /opt/thp/thehive/files
```

TheHive'ı uygun şekilde yapılandırın:

```
## Ek depolama yapılandırması
storage {
  ## Local filesystem
  provider: localfs
  localfs {
    location: /opt/thp/thehive/files
  }
}
```

Mın.IO:

1. Min.IO kümesi yükleme
2. TheHive'ın her bir düğümünü uygun şekilde yapılandırın.

/etc/thehive/application.conf with TheHive 5.0.x

```
## Attachment storage configuration
storage {
  provider: s3
  s3 {
    bucket = "thehive"
    readTimeout = 1 minute
    writeTimeout = 1 minute
    chunkSize = 1 MB
```

```
    endpoint = "http://<IP_MINIO_1>:9100"
    accessKey = "thehive"
    secretKey = "password"
    region = "us-east-1"
  }
}
alpakka.s3.access-style = path
```

/etc/thehive/application.conf with TheHive > 5.0

```
storage {
  provider: s3
  s3 {
    bucket = "thehive"
    readTimeout = 1 minute
    writeTimeout = 1 minute
    chunkSize = 1 MB
    endpoint = "http://<IP_MINIO_1>:9100"
    accessKey = "thehive"
    aws.credentials.provider = "static"
    aws.credentials.secret-access-key = "password"
    access-style = path
    aws.region.provider = "static"
    aws.region.default-region = "us-east-1"
  }
}
```

- Yapılandırma geriye dönük uyumludur
- MinIO yapılandırmasında hiçbiri belirtilmemişse us-east-1 varsayılan bölgedir. Bu durumda, bu parametre isteğe bağlıdır.

Bağlayıcılar Yapılandırması

TheHive bağlayıcıları

TheHive, Cortex ve MISP ile entegre olmak için bağlayıcılara sahiptir.

Varsayılan olarak, bunlar */etc/thehive/application.conf* yapılandırma dosyasında etkindir. Eğer bunlardan birini veya her ikisini kullanmıyorsanız ilgili satırı yorumlayarak devre dışı bırakabilirsiniz:

/etc/thehive/application.conf

```
[..]
scalligraph.modules += org.thp.thehive.connector.cortex.CortexModule
# scalligraph.modules += org.thp.thehive.connector.misp.MispModule  #
```

> Yapılandırma dosyasının güncellenmesi için TheHive hizmetinin yeniden başlatılması gerekir.

Akka Yapılandırması

Akka, Java ve Scala için yüksek düzeyde eşzamanlı, dağıtılmış ve esnek mesaj odaklı uygulamalar oluşturmaya yönelik bir araç setidir

-- https://akka.io/

Akka, TheHive'in birçok düğümünü bir araya getirerek ve sorunsuz bir kullanıcı deneyimi sunarak çalışmasını sağlamak için kullanılır.

Temel yapılandırma

İyi bir küme kurulumu, TheHive uygulamalarının en az 3 düğümünü gerektirir. Her düğüm için Akka şu şekilde yapılandırılmalıdır:

```
akka {
  cluster.enable = on
  actor {
    provider = cluster
  }
  remote.artery {
    canonical {
      hostname = "<HOSTNAME OR IP_ADDRESS>"
      port = 2551
    }
  }
# seed node list contains at least one active node
  cluster.seed-nodes = [ "akka://application@HOSTNAME1:2551", "akka://application@HOSTNAME2:2551",
"akka://application@HOSTNAME3:2551" ]
}
```

ile:

remote.artery.hostname düğümün ana bilgisayar adını veya IP adresini içerir,
akka düğümlerinin listesini içeren ve tüm düğümlerde aynı olan *cluster.seed-nodes*

Bir Kümenin 3 Düğümlü Yapılandırması:
Düğüm 1:

```
akka {
   cluster.enable = on
   actor {
     provider = cluster
   }
   remote.artery {
     canonical {
        hostname = "10.1.2.1"
        port = 2551
     }
   }
   # seed node list contains at least one active node
   cluster.seed-nodes = [ "akka://application@10.1.2.1:2551", "akka://application@10.1.2.2:2551",
"akka://application@10.1.2.3:2551" ]
}
```

Düğüm 2:

```
akka {
   cluster.enable = on
   actor {
   provider = cluster
   }
   remote.artery {
   canonical {
      hostname = "10.1.2.2"
      port = 2551
   }
   }
   # seed node list contains at least one active node
   cluster.seed-nodes = [ "akka://application@10.1.2.1:2551", "akka://application@10.1.2.2:2551",
"akka://application@10.1.2.3:2551" ]
}
```

Düğüm3:

```
akka {
   cluster.enable = on
   actor {
   provider = cluster
```

```
    }
    remote.artery {
    canonical {
       hostname = "10.1.2.3"
       port = 2551
    }
    }
    # seed node list contains at least one active node
    cluster.seed-nodes = [ "akka://application@10.1.2.1:2551", "akka://application@10.1.2.2:2551",
"akka://application@10.1.2.3:2551" ]
}
```

SSL/TLS desteği

Akka, düğümler arasındaki iletişimi şifrelemek için SSL/TLS'yi destekler. SSL desteği ile tipik bir yapılandırma :

```
akka {
  cluster.enable = on
  actor {
    provider = cluster
  }
  remote.artery {
    transport = tls-tcp
    canonical {
      hostname = "<HOSTNAME OR IP_ADDRESS>"
      port = 2551
    }

    ssl.config-ssl-engine {
      key-store = "<PATH TO KEYSTORE>"
      trust-store = "<PATH TO TRUSTSTORE>"

      key-store-password = "chamgeme"
      key-password = "chamgeme"
      trust-store-password = "chamgeme"

      protocol = "TLSv1.2"
    }
  }
# seed node list contains at least one active node
```

```
  cluster.seed-nodes = [ "akka://application@HOSTNAME1:2551", "akka://application@HOSTNAME2:2551",
"akka://application@HOSTNAME3:2551" ]
}
```

Not: *akka.remote.artery.transport*'un değiştiğini ve *akka.ssl.config-ssl-engine*'in yapılandırılması gerektiğini unutmayın.

Referans: https://doc.akka.io/docs/akka/current/remoting-artery.html#remote-security

Sertifikalar Hakkında

Sertifikalar Hakkında

Kendi dahili PKI'nızı veya sertifikalarınızı oluşturmak için keytool komutlarını kullanabilirsiniz.

Referans: https://lightbend.github.io/ssl-config/CertificateGeneration.html#using-keytool

Sunucu sertifikalarınızın her şeyin düzgün çalışması için çeşitli KeyUsage ve ExtendedkeyUsage uzantıları içermesi gerekir:

- KeyUsage uzantıları
  - nonRepudiation
  - dataEncipherment
  - digitalSignature
  - keyEncipherment
- ExtendedkeyUsage uzantıları
  - serverAuth
  - clientAuth

Node 1 için SSL ile Akka yapılandırması:

```
akka {
  cluster.enable = on
  actor {
    provider = cluster
  }
  remote.artery {
    transport = tls-tcp
    canonical {
      hostname = "10.1.2.1"
      port = 2551
    }

    ssl.config-ssl-engine {
```

```
      key-store = "/etc/thehive/application.conf.d/certs/10.1.2.1.jks"
      trust-store = "/etc/thehive/application.conf.d/certs/internal_ca.jks"

      key-store-password = "chamgeme"
      key-password = "chamgeme"
      trust-store-password = "chamgeme"

      protocol = "TLSv1.2"
    }
  }
# seed node list contains at least one active node
  cluster.seed-nodes = [ "akka://application@10.1.2.1:2551", "akka://application@10.1.2.2:2551",
"akka://application@10.1.2.3:2551" ]
}
```

Aynı prensibi diğer düğümler için de uygulayın ve tüm hizmetleri yeniden başlatın.

Logs (Günlükler) Yapılandırması

TheHive, çalışan süreç hakkında bilgi kaydetmek için logback kullanır. Logs, /etc/thehive/logback.xml dosyasında yapılandırılır. Bu dosyayı düzenleyin ve değişikliklerinizi uygulamak için hizmeti yeniden yükleyin.

Varsayılan olarak, günlükler /var/log/thehive/ dizininde depolanır. Son günlük dosyasına application.log denir ve eski dosyalar application.%i.log.zip biçiminde sıkıştırılmış bir formatta saklanır.

Log Seviyesini Artırma/Azaltma

Logback, birkaç log seviyesini destekler.

Daha fazla şeyi kaydetmek için kök düzeyini DEBUG (veya TRACE) olarak güncelleyebilirsiniz:

logback.xml

```
<!-- ... -->
<root level="DEBUG">
  <!-- ... -->
</root>
```

Daha az şeyi log kaydetmek için WARN, ERROR veya OFF seviyelerini kullanabilirsiniz.

Log seviyesi, belirli bir kaydedicinin seviyesi değiştirilerek ayrı ayrı da güncellenebilir:

logback.xml

```
<logger name="org.thp" level="DEBUG"/>
```

Docker'da Logs

Docker içindeki konteynerde, günlükleyici varsayılan olarak */etc/thehive/logback.xml* dosyasıyla yapılandırılır ve uygulama stdout'a ve */var/log/thehive/application.log*'a günlükler.

Varsayılan yapılandırmayı değiştirmek isterseniz, kendi logback dosyanızı /etc/thehive/logback.xml'ye bağlayabilirsiniz.

Logback Yapılandırmanızı Hata Ayıklama

Eğer sorunlarınız varsa logback.xml'de debug bayrağını true olarak ayarlayın:

logback.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration debug="true">
```

Bu, uygulama başladığında logback yapılandırmanızı konsola kaydeder.

Log Erişimi Nasıl Oluşturulur

Logback yapılandırmasını değiştirerek, belirli günlükleri uygulamadan başka bir yere yönlendirebilirsiniz. Aşağıda, erişim günlüklerinin *access.log* dosyasına yönlendirildiği ve bir döner dosya stratejisi kullandığı bir örnek bulunmaktadır.

Bu yapılandırmayı uygulamak için, ekleyicileri ve günlükleyicilerin tanımlarını kopyalayın.

logback.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration debug="false">

  <!-- ... other appenders and settings -->

  <appender name="ACCESSFILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>/var/log/thehive/access.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy">
      <fileNamePattern>/var/log/thehive/access.%i.log.zip</fileNamePattern>
      <minIndex>1</minIndex>
      <maxIndex>10</maxIndex>
    </rollingPolicy>
    <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
      <maxFileSize>10MB</maxFileSize>
    </triggeringPolicy>

    <encoder>
      <pattern>%date [%level] from %logger [%traceID] %message%n%xException</pattern>
    </encoder>
  </appender>

  <appender name="ASYNCACCESSFILE" class="ch.qos.logback.classic.AsyncAppender">
    <appender-ref ref="ACCESSFILE"/>
  </appender>
```

```
    <logger name="org.thp.scalligraph.AccessLogFilter">
        <appender-ref ref="ASYNCACCESSFILE" />
    </logger>
    <logger name="org.thp.scalligraph.controllers.Entrypoint">
        <appender-ref ref="ASYNCACCESSFILE" />
    </logger>

    <root level="INFO">
        <!-- other appender-refs ... -->
    </root>

</configuration>
```

Log'lar syslog'a Nasıl Gönderilir

Bir SyslogAppender eklemeniz gerekecektir.

logback.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration debug="false">

    <!-- ... other appenders and settings -->

    <appender name="SYSLOG" class="ch.qos.logback.classic.net.SyslogAppender">
        <syslogHost>remote_host</syslogHost>
        <facility>AUTH</facility>
        <suffixPattern>[%thread] %logger %msg</suffixPattern>
    </appender>

    <root level="INFO">
        <appender-ref ref="SYSLOG" />
        <!-- other appender-refs ... -->
    </root>
```

Sınırlamalar: Resmi syslog uygulayıcısı günlükleri yalnızca UDP üzerinden bir sunucuya gönderebilir. TCP ve TLS kullanamaz

Proxy Yapılandırması

Proxy Ayarları

Genel Uygulama için Proxy

Proxy kullanılabilir. Varsayılan olarak, JVM'de yapılandırılan proxy kullanılır, ancak her HTTP istemcisi için özel yapılandırmalar yapılabilmektedir.

Parameter	Type	Description
`wsConfig.proxy.host`	string	The hostname of the proxy server
`wsConfig.proxy.port`	integer	The port of the proxy server
`wsConfig.proxy.protocol`	string	The protocol of the proxy server. Use "http" or "https". Defaults to "http" if not specified
`wsConfig.proxy.user`	string	The username of the credentials for the proxy server
`wsConfig.proxy.password`	string	The password for the credentials for the proxy server
`wsConfig.proxy.ntlmDomain`	string	The NTLM domain
`wsConfig.proxy.encoding`	string	The realm's charset
`wsConfig.proxy.nonProxyHosts`	list	The list of hosts on which proxy must not be used

Secrets (Gizli) Yapılandırma

secret.conf dosyası

Bu dosya, kullanıcı oturumunu yönetmek için kullanılan çerezleri tanımlamak için kullanılan bir sırrı içerir. Sonuç olarak, TheHive'ın bir örneği benzersiz bir sır anahtarı kullanmalıdır.

Örnek:

```
play.http.secret.key="dgngu325mbnbc39cxas4l5kb24503836y2vsvsg465989fbsvop9d09ds6df6"
```

> TheHive düğümlerinden oluşan bir küme söz konusu olduğunda, tüm düğümler aynı gizli anahtarla aynı secret.conf dosyasına sahip olmalıdır. Gizli anahtar kullanıcı oturumları oluşturmak için kullanılır.

SSL Yapılandırması

HTTPS kullanarak TheHive'a bağlanın

SSL katmanını yönetmek için bir ters proxy kullanmanızı öneririz; Örneğin, Nginx.

Reference: Configuring HTTPS servers on nginx.org

/etc/nginx/sites-available/thehive.conf

```
server {
  listen 443 ssl http2;
  server_name thehive;

  ssl on;
  ssl_certificate       path-to/thehive-server-chained-cert.pem;
  ssl_certificate_key   path-to/thehive-server-key.pem;

  proxy_connect_timeout   600;
  proxy_send_timeout      600;
  proxy_read_timeout      600;
  send_timeout            600;
  client_max_body_size    2G;
  proxy_buffering off;
  client_header_buffer_size 8k;

  location / {
    add_header             Strict-Transport-Security "max-age=31536000; includeSubDomains";
    proxy_pass             http://127.0.0.1:9000/;
    proxy_http_version      1.1;
  }
}
```

İstemci Yapılandırması

Uzak servislere bağlanmak için SSL yapılandırması gerekebilir. Aşağıdaki parametreler tanımlanabilir:

Parameter	Type	Description

`wsConfig.ssl.keyManager.stores`	list	Stores client certificates (see #certificate-manager)
`wsConfig.ssl.trustManager.stores`	list	Stored custom Certificate Authorities (see #certificate-manager
`wsConfig.ssl.protocol`	string	Defines a different default protocol (see #protocols)
`wsConfig.ssl.enabledProtocols`	list	List of enabled protocols (see #protocols)
`wsConfig.ssl.enabledCipherSuites`	list	List of enabled cipher suites (see #ciphers)
`wsConfig.ssl.loose.acceptAnyCertificate`	boolean	Accept any certificates *true / false*

Sertifika Yöneticisi

Sertifika yöneticisi, istemci sertifikalarını ve sertifika yetkililerini depolamak için kullanılır. Özel Sertifika Yetkilileri Kullanma#

Özel Sertifika Yetkililerini kullanmanın tercih edilen yolu sistem yapılandırmasını kullanmaktır.

Uygulama genelinde özel bir Sertifika Yetkilisi kurulumu gerekiyorsa (web proxy'lerine, LPAPS sunucusu gibi uzak hizmetlere bağlanmak için), daha iyi bir çözüm, bunu işletim sistemine yükleyip TheHive'ı yeniden başlatmaktır.

Debian :

ca-certificates-java paketinin kurulu olduğundan emin olun ve CA sertifikasını doğru klasöre kopyalayın. Ardından dpkg-reconfigure ca-certificates komutunu çalıştırın ve TheHive hizmetini yeniden başlatın.

```
apt-get install -y ca-certificates-java
mkdir /usr/share/ca-certificates/extra
cp mycustomcert.crt /usr/share/ca-certificates/extra
dpkg-reconfigure ca-certificates
service thehive restart
```

RPM :

Fedora veya RHEL'de ek pakete gerek yoktur. CA sertifikasını doğru klasöre kopyalayın, update-ca-trust'ı çalıştırın ve TheHive hizmetini yeniden başlatın.

```
cp mycustomcert.crt /etc/pki/ca-trust/source/anchors
sudo update-ca-trust
service thehive restart
```

Bir alternatif yöntem, ayrı bir güven deposu kullanmaktır; ancak bu Tercih Edilen bir seçenek DEĞİLDİR. TheHive yapılandırmasında trustManager anahtarını kullanın. Bu, uzak bir ana bilgisayarla güvenli bir bağlantı kurmak için kullanılır. Sunucu sertifikası, güvenilir bir sertifika otoritesi tarafından imzalanmış olmalıdır.

```
wsConfig.ssl.trustManager {
  stores = [
    {
      type = "JKS" // JKS or PEM
      path = "keystore.jks"
      password = "password1"
    }
  ]
}
```

İstemci Sertifikaları

keyManager, HTTP istemcisinin (sertifika tabanlı kimlik doğrulaması kullanıldığında) kendini uzak sunucuda kimlik doğrulamak için hangi sertifikayı kullanabileceğini belirtir.

```
wsConfig.ssl.keyManager {
  stores = [
    {
      type = "pkcs12" // JKS or PEM
      path = "mycert.p12"
      password = "password1"
    }
  ]
}
```

Protokoller

Farklı bir varsayılan protokol tanımlamak istiyorsanız, bunu istemcide özel olarak ayarlayabilirsiniz:

```
wsConfig.ssl.protocol = "TLSv1.2"
```

Etkin protokollerin listesini tanımlamak istiyorsanız, bunu açıkça bir liste sağlayarak yapabilirsiniz:

```
wsConfig.ssl.enabledProtocols = ["TLSv1.2", "TLSv1.1", "TLSv1"]
```

Gelişmiş seçenekler

Parolalar

Parola paketleri wsConfig.ssl.enabledCipherSuites kullanılarak yapılandırılabilir:

```
wsConfig.ssl.enabledCipherSuites = [
  "TLS_DHE_RSA_WITH_AES_128_GCM_SHA256",
  "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
  "TLS_DHE_RSA_WITH_AES_256_GCM_SHA384",
  "TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
]
```

Hata ayıklama

Anahtar yöneticisi / güven yöneticisinde hata ayıklamak için aşağıdaki bayrakları ayarlayın:

```
  wsConfig.ssl.debug = {
    ssl = true
    trustmanager = true
    keymanager = true
    sslctx = true
    handshake = true
    verbose = true
    data = true
    certpath = true
  }
```

Hizmet Yapılandırması

Dinleme adresi ve bağlantı noktası

Uygulama varsayılan olarak tüm arayüzleri ve 9000 numaralı portu dinler. Bu, application.conf dosyasında aşağıdaki parametrelerle dinleme adresini ve bağlantı noktalarını belirtmek mümkündür:

```
http.address=127.0.0.1
http.port=9000
```

Bağlam

Ters proxy kullanıyorsanız ve bir konum belirtmek istiyorsanız (örn: /thehive), TheHive yapılandırmasını da güncellemeniz gerekir

```
play.http.context: "/thehive"
```

Akışlar için özel yapılandırma

Nginx gibi bir ters proxy kullanıyorsanız, aşağıdaki mesajı içeren hata pop-up'ları alabilirsiniz: StreamSrv 504 Ağ Geçidi Zaman Aşımı.

Uzun yoklama yenilemesi için varsayılan ayarı değiştirmeniz gerekir, stream.longPolling.refresh'i buna göre ayarlayın.

```
stream.longPolling.refresh: 45 seconds
```

Web proxy kullanımı

Eğer TheHive'ın önünde bir NGINX ters proxy kullanıyorsanız, bu, metin verisi ile bir dosya yükleme arasında ayırım yapmaz.

Bu nedenle, NGINX sunucu yapılandırmanızda, TheHive *application.conf* dosyasında belirtilen dosya yükleme ve metin boyutu arasındaki en yüksek değere eşit olan *client_max_body_size* parametresini ayarlamanız gerekir.

Uyumluluk

GDPR

> Bu özellik, THeHive 5.x'in Platinum planı ile yalnızca mevcuttur.

TheHive, veritabanındaki veri saklama politikasını yönetmeye yönelik bir özelliğe sahiptir, bu özellik varsayılan olarak etkin değildir.

Stratejiler

İki strateji mevcuttur:

1. Hassas değerleri <redacted> ile değiştirin
2. Verileri silme

Değerleri <redacted> ile Değiştirin

- Vakalar için aşağıdaki alanlar redakte edilir:

davanin `summary` ve `message`
yorumların `message`
görev günlüklerinde `message`
`gdpr.dataTypesToDelete` yapılandırma özelliğinde seçilen ve doldurulan veri tipleri için gözlemlenebilirlerin `message`
sayfaların `content`
TTP'lerdeki prosedürlerin `description`

- Uyarılar için aşağıdaki alanlar redakte edilir:

uyarı `message`
gözlemlenebilirlerin `message` (bkz. `gdpr.dataTypesToDelete` yapılandırma özelliği)
prosedürlerin `description` (ttp)

Denetimler İçin:

alan `details` redakte edilmiştir.

Verileri Sil

Strateji `delete` seçilirse:

Vaka ve bileşenleri - *tasks, task logs, procedures, comments, pages, custom events in timelines, values of custom field and observables* - are deleted
Uyarı ve bileşenleri- *procedures, comments, values of custom field and observables* - are deleted.
Denetim silinir

Saklama

RetentionPeriod parametresi, silinecek veya redakte edilecek verilerin minimum yaşını tanımlar. GDPR süreci, bu ayardan daha eski verilere uygulanacaktır. Yaş, son güncelleme tarihine (veya hiç güncellenmemişse oluşturma tarihine) dayanır. Biçim bir sayı ve bir zaman birimidir. Desteklenen birimler şunlardır:

gün: `d`, `day`
saat: `h`, `hr`, `hour`
dakika: `m`, `min`, `minute`
saniye: `s`, `sec`, `second`
milisaniye: `ms`, `milli`, `millisecond`

Yapılandırma

Bunu etkinleştirmek için /etc/thehive/application.conf yapılandırma dosyası güncellenmelidir. Aşağıdaki yapılandırmayı dosyaya ekleyin:

```
gdpr {
  enabled = true

  ## Format http://www.quartz-scheduler.org/documentation/quartz-2.3.0/tutorials/crontrigger.html
  ## Every Sunday at 02:30

  schedule = "0 30 2 ? * SUN"

  ## Possible GDPR strategies:
  ##   delete: remove the documents
  ##   redact: replace sensitive values by "<redacted>" (cf. dataTypesToDelete)

  strategy = "delete"

  ## if the strategy is "redacted", the observable with dataType in
  ## "dataTypesToDelete" will be removed
  ## for other observables, message will be "<redacted>", not the data
  ## Uncomment following line to select datatypes

  # dataTypesToDelete = [] ## ["ip", "domain"]
```

```
    ## only documents older than the "retentionPeriod" will be processed

    retentionPeriod = 730 days # 2 years

    ## Advanced parameters (should not be modified)

    jobTimeout = 24 days ## maximum time the job is executed
    batchSizeCase = 5     ## how many cases is processed per transaction
    batchSizeAlert = 10   ## how many cases is processed per transaction
    batchSizeAudit = 100  ## how many cases is processed per transaction
}
```

Ardından uygulamayı yeniden başlatın.