



# **SANS Institute**

## Information Security Reading Room

### **Data Mining in the Dark: Darknet Intelligence Automation**

---

Brian Nafziger

Copyright SANS Institute 2021. Author Retains Full Rights.

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

# Data Mining in the Dark:

## Darknet Intelligence Automation

Author: Brian Nafziger, [brian@nafziger.net](mailto:brian@nafziger.net)

Advisor: Johannes B. Ullrich, Ph.D.

Accepted: November 17<sup>th</sup>, 2017

### Abstract

Open-source intelligence offers value in information security decision making through knowledge of threats and malicious activities that potentially impact business. Open-source intelligence using the internet is common, however, using the darknet is less common for the typical cybersecurity analyst. The challenges to using the darknet for open-source intelligence includes using specialized collection, processing, and analysis tools. While researchers share techniques, there are few publicly shared tools; therefore, this paper explores an open-source intelligence automation toolset that scans across the darknet - connecting, collecting, processing, and analyzing. It describes and shares the tools and processes to build a secure darknet connection, and then how to collect, process, store, and analyze data. Providing tools and processes serves as an on-ramp for cybersecurity intelligence analysts to search for threats. Future studies may refine, expand, and deepen this paper's toolset framework.

# 1. INTRODUCTION

Open-source intelligence uses public data to derive some actionable value for use in decision making. The Intelligence Threat Handbook defines intelligence as "the product resulting from the collection, collation, evaluation, analysis, integration, and interpretation of collected information" (IOSS, 1991; IOSS, 1996).

Open-source intelligence utilizing the internet brings value to communities such as business, government, military, and law enforcement for tasks such as decision-making and finding current and future threats. An example of value is monitoring the internet for terrorist and hacktivist threats (Conway, 2006). Open-source intelligence which uses the internet is challenging: collecting is complex (specialized software, authentication), processing is complex (translation, extracting, tagging, and indexing), and analyzing is complex (relational linking) (Best, 2011). However, open-source intelligence which utilizes the internet shows increasing maturity and sophistication with numerous publicly available intelligence tools, such as Maltego and Recon-NG. (Layton, & Watters, 2015)

The internet encompasses a broad set of data. The clear net layer is a subset of internet data indexed by search engines. The deep net layer is a subset of internet data not indexed by search engines. And the darknet layer is a subset of deep net data that is not indexed by search engines and requires specialized software to access, such as The Onion Router (TOR) (Ciancaglini, Balduzzi, McArdle, & Rösler, 2015). The darknet is of value due to its anonymous content hosting which often encourages criminal elements sharing of information, and therefore it can be a valuable area for finding current and future threats. (van den Brink, Broekman, Rijken, Oggero, & Verburgh, 2016). While these publicly available tools, as mentioned earlier, have matured or are maturing into the clear and deep net, these intelligence tools have not matured into the darknet which is the focus of this paper (Nunes, Diab, Gunn, Marin, Mishra, Paliath, & Shakarian, 2016).

Various communities and researchers share techniques for traversing the darknet including Zhang's "Developing a Dark Web Collection Infrastructure", Sanchez-Rola's "The Onions Have Eyes" and Butler's "REAPER: an automated, scalable solution for mass credential harvesting" (Zhang, Zeng, Huang, Fan, Yu, Dang, & Chen, 2010; Sanchez-Rola, Balzarotti, & Santos, 2017; Butler, Wardman, & Pratt, 2016). However, there are few

public tools available (LeVeque, 2013). Therefore, there is limited cybersecurity analyst research on the darknet as a source of intelligence production. The lack of tools raises the question, what tools can be built, and shared for automation of darknet intelligence tradecraft? This paper will explore existing tools and or build new tools as needed. And finally, this paper will document the tools, and as applicable, the tactics, techniques, and procedures for automation of the darknet.

## **2. Lighting a Match in the Dark**

### **2.1. Defining the Dark**

Darknet intelligence tradecraft, at the highest level, requires an understanding of a variety of building block concepts. These concepts include the darknet, an anonymity system to access the darknet, and the intelligence lifecycle to collect, process, and analyze data from the darknet. This is the first step in defining the concepts that will form the basis of the toolset.

The darknet, per Moore and Rid's "Cryptopolitik and the Darknet," originates in the five components of human communication recreated in electronic form: privacy, authentication, anonymity, cash, and hidden exchanges (Moore, & Rid, 2016). First is privacy, the act of hiding communications content. Diffie and Hellman's pioneering of public-key encryption and sharing of keys brought ease of digital encryption (Diffie, & Hellman, 1976). Secondly is authentication, the act of confirming communications ownership. Rivest, Shamir, and Adleman's implementation of public-key encryption brought ease of digital encryption with digital signatures (Rivest, Shamir, & Adleman, 1978).

The third component of the darknet is anonymity, the act of hiding communication ownership. Chaum's concept of layering of communication for anonymity served as the influence for "onion routing" (Chaum, 1981). Chaum's influence, Syverson's research, and Dingledine and Mathewson's design efforts, lead to the creation of an onion router known as TOR (Dingledine, Syverson, 2002; Dingledine, Mathewson, & Syverson, 2004). TOR provides within itself end-to-end encryption, authentication, and anonymous web

browsing. Fourthly is cash, the act of hiding the funding of transactions. Chaum's concept of transactions based on blind signatures served as the inspiration for anonymous digital currency (Chaum, 1985). Through Chaum's inspiration, the pseudonymous Satoshi Nakamoto posted a paper to a mailing list detailing methods of using "a peer-to-peer network to generate a system for electronic transactions without relying on trust" (Nakamoto, 2008). As titled by Nakamoto, "bitcoins" brought ease of anonymous digital currency. Bitcoin enthusiasts have long understood bitcoins are at best pseudonymous, and the shift is once again underway to a newer anonymous digital currency such as Dash, Zerocoin, or Monero (Cimpanu, 2017).

The fifth, and last, component of the darknet is hidden exchanges where users host anonymous content, including discussion forums, using encrypted, authenticated, and anonymous communications. Anderson's "Eternity Service" paper served as the stimulus for TOR hidden services (Anderson, 1996; Moore, & Rid, 2016). However, it was several Microsoft researchers who described and popularized the word "darknet" (Biddle, England, Peinado, & Willman, 2002). TOR as previously mentioned, not only provides within itself end-to-end encryption, authentication, and anonymous web browsing but also provides anonymous content hosting known as onion sites. It is this anonymous content hosting where forums discuss worms, botnet, zero-days, and other malicious activities such as insider trading and hacktivism (Fachkha, 2015). Altogether the five components of communication - privacy, authentication, anonymity, cash, and hidden exchanges are the plumbing of the darknet.

Progressing from the darknet concepts to accessing the darknet requires an understanding of anonymity system concepts. Anonymity, per Edman and Yener's "Survey of Anonymous Communication Systems," is simply an adversary observing the senders and recipients in a network and being "unable to discern who is communicating with whom." Anonymity systems provide "unlinkability" between the sender and the recipients and between the receiver and the senders (Edman, & Yener, 2009). The design of anonymity systems falls into two general classifications: high-latency anonymity systems and low-latency anonymity systems. High-latency anonymity systems are for message-based applications that tolerate a delay, such as an email. Low-latency anonymity systems are for real-time applications, such as web browsing (Edman, & Yener, 2009). The simplest

of low-latency anonymity systems are Virtual Private Networks (VPNs). VPNs hide a user's internet destinations from internet service providers (ISP) and governments. They also hide a user's information from destinations on the internet. VPN usage shifts trust from ISP and the government to the VPN provider, providing privacy (Schanzenbach, 2014). The "king" of low-latency anonymity systems is TOR, as previously discussed, providing within itself end-to-end encryption, authentication, anonymous web browsing, and anonymous content hosting (The Guardian, 2013). Sophisticated anonymity requires further understanding of adversaries, threats, threat models, and compartmentalization and isolation. The Appendix and References list a few ideas on increased anonymity.

Continuing from the anonymity system concepts to collecting and processing data requires an understanding of the intelligence lifecycle concepts. Krizan's "Intelligence Essentials for Everyone" defines and simplifies the intelligence lifecycle as Needs, Collection, Processing, Analysis, and Production. As it pertains to building tools, the primary focus of "collecting" is on exploiting open-source information, the focus of "processing" is on the techniques to transform raw data into intelligence information, and the focus of "analyzing" is the interpretation of events for decisions (Krizan, 1999). Furthermore, "analyzing" entails breaking down problems into smaller problems and the examination of related items to determine "the extent to which they confirm, supplement, or contradict each other" (Mathams, 1988). That is, analyzing involves establishing relationships.

Refining the intelligence lifecycle concepts, the "Open Source Intelligence. An Oxymoron or Real Intelligence?" article asserts that open-source intelligence collection, analysis, and production requires toolsets that scrape data and mine data, and that it requires an understanding of big data (Clarke, 2015). Benavides' comprehensive "Open Source Intelligence (OSINT) ToolKit On The Go" suggests the opportunities to be gained using predictive intelligence such as big data, and ponders the potential predictability of emerging threats using big data. Benavides also notes the potential value of event extraction analysis using natural language processing or NLP (Benavides, 2016).

Using the refined intelligence lifecycle concepts, first, the collection tool will collect data using a web crawler and scraper on the darknet. Najork's "Web Crawler

Architecture" simply defines a web crawler as selecting a website from a set, downloading the web pages, extracting the links contained within, and adding those links to the set of websites to visit (Najork, 2009). Mitchell's "Web Scraping with Python" further defines web scraping as extracting specific data from the websites. (Mitchell, 2015). Secondly in the lifecycle, the processing tool will process data using a big data system. Ward & Barker's "Undefined by Data: A Survey of Big Data Definitions" simply terms big data as the storage and analysis of large and or complex data sets using a variety of techniques (Ward, & Barker, 2013). Thirdly in the lifecycle, the analytics tool will explore data, in this case, using NLP and relationship analysis. Liddy's "Natural Language Processing" work defines NLP as a range of techniques for analyzing naturally occurring texts to achieve language processing for a range of tasks (Liddy, 2001). Ingersoll, Morton, & Farris' "Taming Text" book outlines practical NLP examples for named entity recognition (NER) or finding people, places, and things in a text (Ingersoll, Morton, & Farris, 2013). McCue's "Data Mining and Predictive Analysis" book expresses link analysis as determining relationships between people, places, and things (McCue, 2014). Finally in the lifecycle, the production is analogous to data mining. Data mining is the extraction of knowledge from large-scale data, not the extraction of data (Kosala, & Blockeel, 2000; Chakrabarti, Ester, Fayyad, Gehrke, Han, Morishita, & Wang 2006). Therefore, the proper use of data mining in the context of this paper is when the analyst extracts usable knowledge from the toolset.

## 2.2. Connecting to the Dark

The creation of the toolset begins with the choice of a tool for each of the previously defined concepts. As previously discussed the concepts are the anonymity system, the collection using a web crawler and scraper system, the processing using a big data system, and the analysis using a natural language processing (NLP) system and a relational linking system. This is the next step of finding single tools to weave together into a toolset.

The anonymity system is the foundation of the toolset. The anonymity requirements are minimal: a secure host, privacy as offered by a VPN, and anonymous access to onion sites on the darknet as offered by TOR. Ubuntu Linux as a host is free, securable, easy-to-use, and supports the remaining requirements which are a VPN, TOR, web crawling and scraping, big data, NLP NER, and link analysis. Nord VPN supports anonymous payments,

high bandwidth, low latency, high throughput, multiple simultaneous connections, and exists in multiple countries (Mimir, n.d.). The TOR service stands alone as offered by the Tor Project. The anonymity system forms the first layer of the toolset.

Intelligence collection within the toolset includes data extraction using crawling and scraping. The Ahmia Scrapy crawler/scrapper is a free and scriptable engine that offers a solid darknet foundation and allows for modification to the crawling, scraping, and processing of data (Ahmia, 2016). The Ahmia Scrapy crawler/scrapper forms the second layer of the toolset.

Intelligence processing within the toolset includes data ingestion and storage using a big data system. Elasticsearch database is a free and programmable big data system based on Apache Lucene (Elasticsearch, 2010). The Elasticsearch database forms the third layer of the toolset.

Finally, intelligence analysis within the toolset provides visual analysis; NLP to identify people, places, and things; and link analysis to determine relationships between people, places, and things. Elasticsearch offers a visualization plugin called Kibana (Kibana, 2010). The Elasticsearch community offers an NLP NER (Named Entity Recognition) processor based on Apache OpenNLP which detects entities such as names, dates, and locations (Reelson, 2016). Both tools are free and programmable. And finally, Maltego is a free (for personal use) and scriptable link analysis system (Maltego, 2007). Kibana and Maltego form the fourth and final layer of the toolset.

### **2.3. Stepping into the Dark**

The assembly of the toolset now begins. The assembly includes the processes of installing and configuring the anonymity system, the web crawler and scraper system, the big data system, the natural language processing (NLP) system, and the relational linking system. This is the next step of weaving the tools together.

The toolset base system requirements are minimal - hardware, operating system, identity, privacy service - VPN, anonymity service - TOR, processing - Elasticsearch, and analysis service - Maltego. The Appendix and various internet sites describe base implementation builds. Additionally, the base anonymity system does not offer



sophisticated anonymity which needs an understanding of adversaries, threats, threat models, and compartmentalization and isolation. The Appendix and References list a few ideas on increased anonymity.

Once the toolset base implementations are complete, the analyst should continue building the intelligence system analysis service which interprets incoming darknet data using the OpenNLP NER Processor. First, install the prerequisite SDKMAN and Gradle, then install the processor. Now compile and install the processor into Elasticsearch. Finally, restart Elasticsearch. Once the Elasticsearch OpenNLP Processor is running, confirm it is functioning by using curl or by visually browsing to the service. Figure 1 depicts the process.

The Elasticsearch OpenNLP NER Processor can detect entities in text such as names, dates, and locations. To detect entities, it needs a model of what to detect. The Apache OpenNLP project offers several pre-trained NER models which are freely available and the OpenNLP NER Processor packages these models (Reelson, 2016). The Appendix lists the Elasticsearch OpenNLP NER code repository. The code repository also integrates code revisions for OpenNLP DOCCAT (Document Categorizer) processing which classifies text into categories. To classify a text, DOCCAT needs a model as well. However, these models are unique to the task, and there are no pre-trained models. The Appendix lists the revised Elasticsearch OpenNLP NER with DOCCAT code repository and the sample creation and usage of the code revisions with a small generic model.

```
INSTALL GIT Prerequisite

$ sudo apt-get update
$ sudo apt-get install git

INSTALL SDKMAN Prerequisite

$ sudo apt-get install zip
$ curl -s "https://get.sdkman.io" | bash
$ source "/home/guest/.sdkman/bin/sdkman-init.sh"

INSTALL Gradle Prerequisite

$ sdk install gradle 4.1
$ gradle -v
```

```

COMPILE OpenNLP Processor

$ git clone https://github.com/bnafziger/elasticsearch-ingest-
opennlp.git

$ cd /home/guest/Desktop/code/elasticsearch-ingest-opennlp/

$ gradle build
$ gradle clean check

BUILD SUCCESSFUL in 29s
32 actionable tasks: 30 executed, 2 up-to-date

INSTALL OpenNLP Processor

$ sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install
file:///home/guest/Desktop/code/elasticsearch-ingest-
opennlp/build/distributions/ingest-opennlp-5.5.1.1-SNAPSHOT.zip

[=====] 100%
-> Installed ingest-opennlp

$ sudo tee --append /etc/elasticsearch/elasticsearch.yml <<-EOF
ingest.opennlp.model.file.persons: en-ner-persons.bin
ingest.opennlp.model.file.dates: en-ner-dates.bin
ingest.opennlp.model.file.locations: en-ner-locations.bin
EOF

VALIDATE OpenNLP Processor

$ sudo systemctl restart elasticsearch

$ sudo tail /var/log/elasticsearch/elasticsearch.log
[2017-08-15T00:52:04,670][INFO ][o.e.p.PluginsService ] [WCCCQsy]
loaded plugin [ingest-opennlp]
[2017-08-15T00:52:40,356][INFO ][d.s.e.i.o.OpenNlpService ] [WCCCQsy]
Read models in [29.2s] for [dates, locations, persons]

```

**Figure 1 - OpenNLP Processor Build**

Once the Elasticsearch OpenNLP Processor is running, confirm it is properly processing data. First, create a sample database and then create a connection to the OpenNLP Processor service using an Elasticsearch pipeline. Finally, insert data into the database using the pipeline. The queried data show the detected NLP entities in the text such as names, dates, and locations. To visually query requires configuring an index in Kibana and it shows the same entities. Figure 2 describes the process.

```

CREATE Elasticsearch Database

```

```
$ curl -X PUT -i "localhost:9200/my-index/" -d '
{
  "mappings": {
    "my-type": {
      "properties": {
        "my_field": { "type": "string" }
      }
    }
  }
}'
```

#### CREATE Elasticsearch OpenNLP Pipeline

```
$ curl -X PUT localhost:9200/_ingest/pipeline/opennlp-pipeline -d ' {
"description": "A pipeline to do named entity extraction",
"processors": [ { "opennlp" : { "field" : "my_field","ignore_missing":
true } } ] } '
```

#### PUT data into Elasticsearch Database using OpenNLP Pipeline


```
$ curl -X PUT 'localhost:9200/my-index/my-type/1?pipeline=opennlp-
pipeline' -d ' { "my_field" : "Kobe Bryant was one of the best
basketball players of all times. Not even Michael Jordan has ever
scored 81 points in one game. Munich is really an awesome city, but
New York is as well. Yesterday has been the hottest day of the year."
} '
```

#### GET data from Elasticsearch Database

```
$ curl -X GET 'localhost:9200/my-index/my-type/1'
{"_index":"my-index","_type":"my-
type","_id":"1","_version":1,"found":true,"_source":{"my_field":"Kobe
Bryant was one of the best basketball players of all times. Not even
Michael Jordan has ever scored 81 points in one game. Munich is really
an awesome city, but New York is as well. Yesterday has been the
hottest day of the year.","entities":{"persons":["Kobe
Bryant","Michael
Jordan"],"dates":["Yesterday"],"locations":["Munich","New York"]}}}
```

#### SEARCH data using Kibana

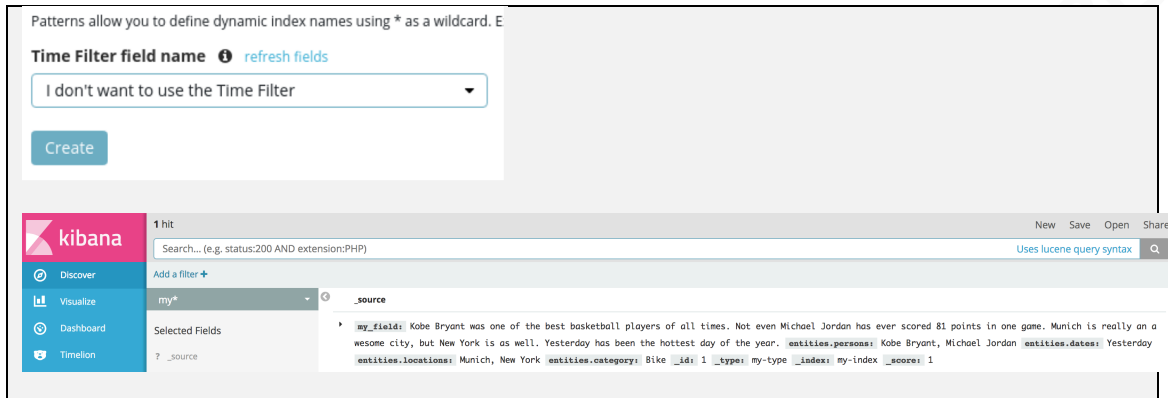
http://localhost:5601

 Management

### Configure an index pattern

In order to use Kibana you must configure at least one index pattern against an Elasticsearch Index to run search and analytics against. They are used to filter and aggregate data.

**Index name or pattern**



**Figure 2 - OpenNLP Processor Testing (Reelson, 2016)**

The analyst should continue building the intelligence system with the collection service which extracts data from the darknet using the Ahmia Scrapy services. Ahmia Scrapy extracts data using TOR and stores data into Elasticsearch using the Elasticsearch OpenNLP Processor pipeline. The Ahmia Scrapy service requires the creation of the database and then the download and installation of a revised Ahmia crawler. The revised Ahmia crawler inserts data into and through the previously created Elasticsearch OpenNLP Processor pipeline. Figure 3 shows the process. The Appendix lists the revised Ahmia Scrapy code repository.

#### DOWNLOAD Ahmia Elasticsearch Database

```
$ git clone https://github.com/bnafziger/ahmia-index.git
$ cd /home/guest/Desktop/code/ahmia-index/
```

#### CREATE Ahmia Elasticsearch Database

```
$ curl -XPUT -i "localhost:9200/crawl/" -d "@./mappings.json"

$ sudo apt-get install build-essential python-pip
$ pip install bs4 requests
$ crontab -e
0 22 * * * cd /home/guest/Desktop/code/ahmia-index/ && torsocks python
child_abuse_onions.py > filter_these_domains.txt && bash
call_filtering.sh
```

#### DOWNLOAD Ahmia Scrapy Crawler and Scraper

```
$ git clone https://github.com/bnafziger/ahmia-crawler.git
$ cd /home/guest/Desktop/code/ahmia-crawler/
```

#### INSTALL Ahmia Scrapy Crawler and Scraper

```
$ sudo apt-get install build-essential python-pip python-virtualenv
```

```

$ sudo apt-get install libxml2-dev libxslt1-dev python-dev libffi-dev
libssl-dev

$ virtualenv ./virtual
$ source virtual/bin/activate
(virtual) $ pip install -r requirements.txt
(virtual) $ deactivate

CREATE Elasticsearch OpenNLP Pipeline

$ curl -X PUT localhost:9200/_ingest/pipeline/opennlp-pipeline -d ' {
"description": "A pipeline to do named entity extraction",
"processors": [ { "opennlp" : { "field" : "content", "ignore_missing":
true } } ] } '

SHOW Ahmia Scrapy Options

$ cd /home/guest/Desktop/code/ahmia-crawler/ahmia
$ scrapy crawl ahmia-tor
-s DEPTH_LIMIT=1
-s LOG_LEVEL=INFO
-s ROBOTSTXT_OBEY=0
-s ALLOWED_DOMAINS=/home/user/allowed_domains.txt
-s TARGET_SITES=/home/user/seed_list.txt
-s ELASTICSEARCH_TYPE=targetitemtype

# stop a spider safely at any time by pressing Ctrl-C

# start a spider with persistence supported enabled
$ scrapy crawl spidername -s JOBDIR=crawls/spidername -l

# stop the spider safely at any time and resume it later
$ scrapy crawl spidername -s JOBDIR=crawls/spidername -l

# stop the spider after number of responses crawled
$ scrapy crawl spidername -s CLOSESPIDER_PAGECOUNT=100

```

**Figure 3 - Ahmia Scrapy Install**

With the Ahmia Scrapy service installed, confirm that it is running and processing data using curl or by visually browsing to the service. The queried data shows the darknet data and the detected NLP entities in the darknet data. To visually query, once again requires configuring an index in Kibana. Figure 4 shows the process.

```

RESTART TOR and Polipo Services

$ sudo systemctl restart tor.service
$ sudo systemctl restart polipo

$ curl --proxy localhost:8123 -A json 'https://ip-show.com'

RUN Ahmia Scrapy Crawler and Scraper

```

```
$ cd /home/guest/Desktop/code/ahmia-crawler
$ source virtual/bin/activate
$ cd /home/guest/Desktop/code/ahmia-crawler/ahmia


(virtual) $ scrapy crawl ahmia-tor -s DEPTH_LIMIT=1 -s
CLOSESPIDER_PAGECOUNT=100
2017-08-16 00:18:17 [scrapy.utils.log] INFO: Scrapy 1.4.0 started
(bot: ahmia)
...
(virtual) $ deactivate
```

GET data from Elasticsearch Database

```
$ curl -X GET 'http://localhost:9200/crawl/_search?q=*&pretty'
{
  "took" : 105,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "failed" : 0
  },
  "hits" : {
    "total" : 393,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "crawl",
        "_type" : "tor",
        "_id" : "fd7d9b3aea3f3a2d7d94faa0e40c7402a67952f2",
        "_score" : 1.0,
        "_source" : {
          "raw_title" : "Daniel - Onion link list",
          "domain" : "onionsnjajzkhm5g.onion",
          "raw_url" : "http://onionsnjajzkhm5g.onion/onions.php",
```

SEARCH using Kibana

http://localhost:5601

 Management

### Configure an index pattern

In order to use Kibana you must configure at least one index pattern to search and analytics against. They are also used to configure filters.

**Index name or pattern**

Patterns allow you to define dynamic index names using \* as a wildcard. E.g. crawl-\* will match all indices starting with crawl-.

**Time Filter field name** [refresh fields](#)

[Create](#)

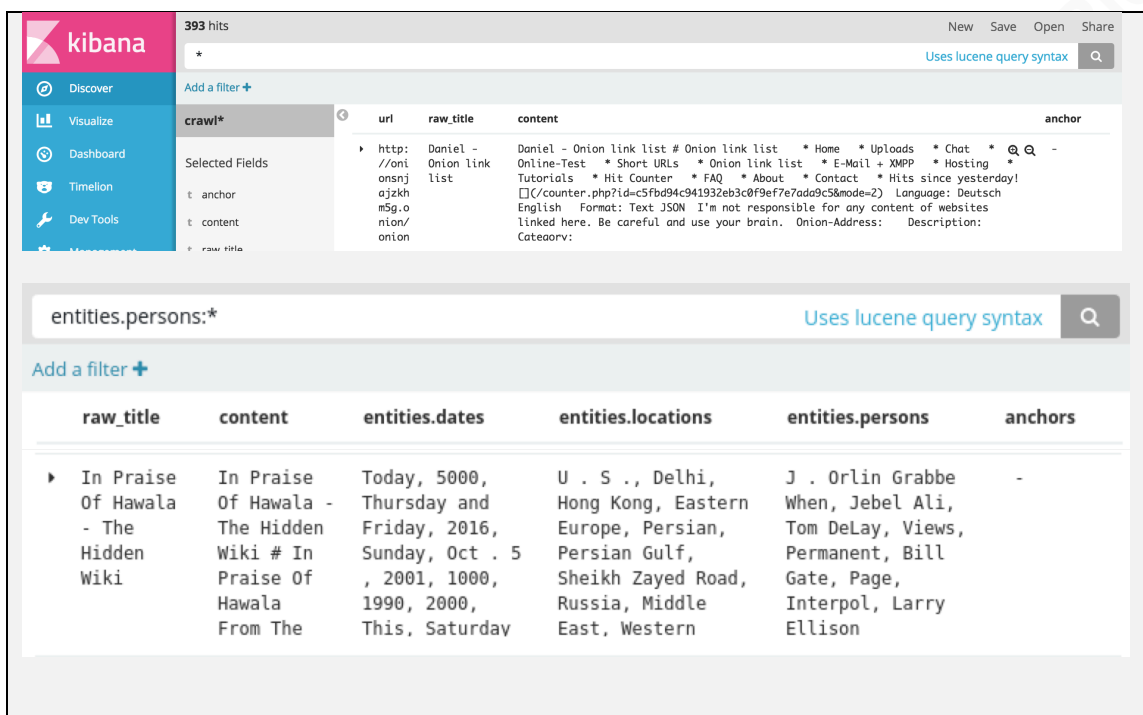


Figure 4 - Ahmia Scrapy Run

The analyst should complete the intelligence system link analysis service by establishing darknet data relationships using Maltego. Maltego requires minimal effort to download and install. First, download and install the Maltego application and then download and install the libraries to support integrating Elasticsearch with Maltego using transforms. Maltego also allows manual command line testing of transforms. Figure 5 shows the process and sample transforms. The Appendix lists the Maltego transform code repository.

Maltego transforms, or scripts, connect a variety of data sources. Using the Elasticsearch API and the Maltego API, it is simple to connect both services using a transform. The transform receives an input entity from the Maltego graphical interface, searches Elasticsearch for the entity, and finally returns the found entities. As an example of a simple case, Maltego can search Elasticsearch for a location entity on the Maltego graphical interface and then return name entities relationships with the location entity.

DOWNLOAD AND INSTALL Maltego

```
$ cd /home/guest/Desktop/code
```

```

$ wget
https://www.paterva.com/malv4/community/MaltegoCE.v4.0.11.9358.deb
$ shasum MaltegoCE.v4.0.11.9358.deb
02be9645a05f203a27e8552b033fddfc7c1af203
$ sudo apt-get install ./MaltegoCE.v4.0.11.9358.deb

INSTALL Maltego Transforms Libraries and Elasticsearch Libraries

$ wget https://docs.paterva.com/media/downloads/MaltegoTransform-
Python.zip
$ unzip -j MaltegoTransform-Python.zip MaltegoTransform-
Python/Maltego*/Maltego* -d ./MaltegoTransform-Python
$ cd /home/guest/Desktop/code/MaltegoTransform-Python

$ pip install elasticsearch

DOWNLOAD Maltego Transforms

$ git clone https://github.com/bnafziger/MaltegoTransform-Python.git
$ cd /home/guest/Desktop/code/MaltegoTransform-Python/

OR BUILD Maltego Transforms

$ tee ./ToURL-UsingElasticsearchLocationQuery.py <<-EOF
#!/usr/bin/env python

from elasticsearch import Elasticsearch
from MaltegoTransform import *
import os

phrase = sys.argv[1]
m = MaltegoTransform()

try:
    es = Elasticsearch('http://127.0.0.1:9200')
    res = es.search(index="crawl", doc_type="tor", body={"query":
{"match": {"entities.locations": phrase}}})
    for doc in res['hits']['hits']:
        m.addEntity('maltego.URL', doc['_source']['url'])

except Exception as e:
    m.addUIMessage(str(e))

m.returnOutput()
EOF

$ tee ./ToPerson-UsingElasticsearchURLQuery.py <<-EOF
#!/usr/bin/env python

from elasticsearch import Elasticsearch
from MaltegoTransform import *
import os

phrase = sys.argv[1]
m = MaltegoTransform()

```



```

try:
    es = Elasticsearch('http://127.0.0.1:9200')
    res = es.search(index="crawl", doc_type="tor", body={"query":
{"match": {"url": phrase}}})
    for doc in res['hits']['hits']:
        m.addEntity('maltego.Person', ', ',
'.join(doc['_source']['entities']['persons']).decode('utf-8',
'ignore'))

except Exception as e:
    m.addUIMessage(str(e))

m.returnOutput()
EOF

$ tee ./ToLocation-UsingElasticsearchURLQuery.py <<-EOF
#!/usr/bin/env python

from elasticsearch import Elasticsearch
from MaltegoTransform import *
import os

phrase = sys.argv[1]
m = MaltegoTransform()

try:
    es = Elasticsearch('http://127.0.0.1:9200')
    res = es.search(index="crawl", doc_type="tor", body={"query":
{"match": {"url": phrase}}})
    for doc in res['hits']['hits']:
        m.addEntity('maltego.Location', ', ',
'.join(doc['_source']['entities']['locations']).decode('utf-8',
'ignore'))

except Exception as e:
    m.addUIMessage(str(e))

m.returnOutput()
EOF

$ tee ./ToURL-UsingElasticsearchMLTQuery.py <<-EOF
#!/usr/bin/env python

from elasticsearch import Elasticsearch
from MaltegoTransform import *
import os

phrase = sys.argv[1]
m = MaltegoTransform()

try:
    es = Elasticsearch('http://127.0.0.1:9200')
    res = es.search(

```

```

index="crawl",
doc_type="tor",
body =
{
  "query":
  {
    "more_like_this" :
    {
      "like" : phrase,
      "min_term_freq" : 0,
      "max_query_terms" : 25
    }
  }
}
})

for doc in res['hits']['hits']:
    ent = m.addEntity('maltego.URL', doc['_source']['url'])
    #ent = m.addEntity('maltego.Person', ' ',
'.join(doc['_source']['entities']['persons']).decode('utf-8',
'ignore'))
    #ent = m.addEntity('maltego.Location', ' ',
'.join(doc['_source']['entities']['locations']).decode('utf-8',
'ignore'))

except Exception as e:
    m.addUIMessage(str(e))

m.returnOutput()
EOF

TEST Maltego Transforms

$ python ToURL-UsingElasticsearchLocationQuery.py "Sheikh Zayed Road"
<MaltegoMessage>
<MaltegoTransformResponseMessage>
<Entities>
<Entity Type="maltego.URL">
<Value>http://zqktlwi4fecvo6ri.onion/wiki/In_Praise_Of_Hawala</Value>
<Weight>100</Weight>
</Entity>
...

$ python ToPerson-UsingElasticsearchURLQuery.py
"http://zqktlwi4fecvo6ri.onion/wiki/In_Praise_Of_Hawala"
<MaltegoMessage>
<MaltegoTransformResponseMessage>
<Entities>
<Entity Type="maltego.Person">
<Value>J . Orlin Grabbe When, Jebel Ali, Tom DeLay, Views, Permanent,
Bill Gate, Page, Interpol, Larry Ellison</Value>
<Weight>100</Weight>
</Entity>
...

$ python ToURL-UsingElasticsearchMLTQuery.py "Hack Forum"
<MaltegoMessage>
<MaltegoTransformResponseMessage>
<Entities>
<Entity Type="maltego.URL">
<Value>http://hellamz4kpl26ltr.onion/</Value>

```

```

<Weight>100</Weight>
</Entity>
...

RUN Maltego

$ /usr/share/MaltegoCommunity/bin/maltego

```

**Figure 5 - Maltego Transform Build**

Once Maltego service is running, confirm it is working by configuring the transform and using the graphical interface. To use Maltego register an account using the pseudo-anonymous identity. Then start Maltego and log on using the account. Using Maltego first requires configuring the transform or transforms. Select New Local Transforms and enter the name, id, and author. For the input entity type use the included examples, select Location for the first example, and URL for the second example and person for the third example. Select no transform set. For the command use /usr/bin/python, and for the parameter use ToURL-UsingElasticsearchLocationQuery.py, ToPerson-UsingElasticsearch URL Query.py, and ToURL-Using Elasticsearch MLT Query.py respectively. Finally, for the working directory use /home/guest/Desktop/code/MaltegoTransform-Python.

Performing a Maltego link analysis now requires a starting entity. Select Create a New Graph and open the Entity Palette. On the palette, expand locations and drag and drop a location onto the graph. Double click and rename the location entity with an actual location. Now right click on the location and run the "To URL - Using Elasticsearch" transform. Running the location transform will return a list of URL's that contain the location. Now right click on a URL and run the "To Person - Using Elasticsearch" transform. Running the URL transform will return a list of persons at that URL. Figure 6 shows the results.

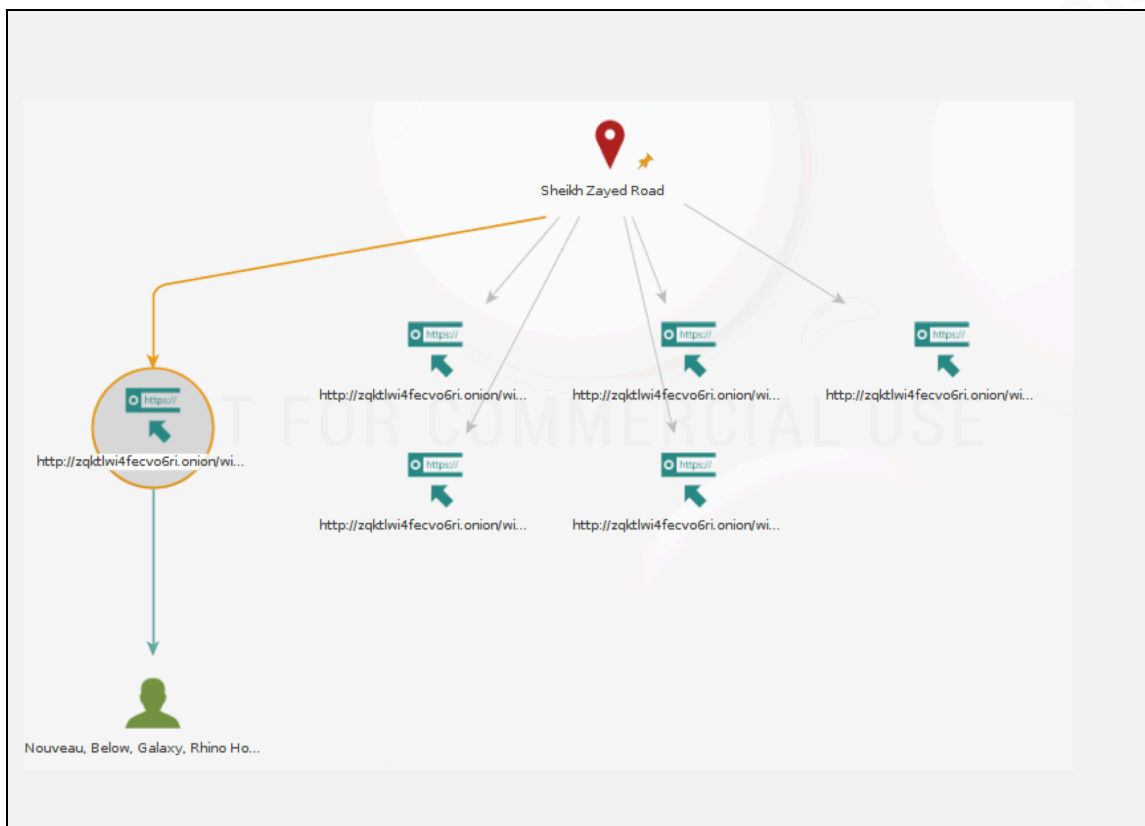


Figure 6 - Maltego Transform Run

## 2.4. Lighting a Match in the Dark

The assembled toolset now offers a complete open-source intelligence process. The process includes running the anonymity system, the web crawler and scraper, the big data system, the natural language processing (NLP) system, and the relational linking system. This is the final step of running the complete toolset.

Start collection and processing by cleaning Elasticsearch database and OpenNLP pipeline. Next, restart the Elasticsearch database system. The VPN privacy service is still running. Now restart the TOR anonymity service. Finally, run the Ahmia Scrapy service. To increase the scraping scope, increase the page count and download and use a recent onion seed list. Figure 7 shows the complete process.

```
DELETE Database
```

```
$ curl -XDELETE "localhost:9200/crawl/"
```

```
$ curl -X GET 'http://localhost:9200/crawl/_search?q=*&pretty'
```

```
DELETE Pipeline
```

```

$ curl -X DELETE "localhost:9200/_ingest/pipeline/opennlp-pipeline/"
$ curl -X GET localhost:9200/_ingest/pipeline/opennlp-pipeline

CREATE Ahmia Database

$ cd /home/guest/Desktop/code/ahmia-index/
$ curl -XPUT -i "localhost:9200/crawl/" -d "@./mappings.json"

CREATE OpenNLP Pipeline

$ curl -X PUT localhost:9200/_ingest/pipeline/opennlp-pipeline -d ' {
"description": "A pipeline to do named entity extraction",
"processors": [ { "opennlp" : { "field" : "content", "ignore_missing":
true } } ] }'

RESTART Elasticsearch Kibana Service

$ sudo systemctl restart elasticsearch
$ sudo systemctl restart kibana.service

RESTART TOR and Polipo Services

$ sudo systemctl restart tor.service
$ sudo systemctl restart polipo
$ curl --proxy localhost:8123 -A json 'https://ip-show.com'

RUN Ahmia Scrapy Services

$ cd /home/guest/Desktop/code/ahmia-crawler/ahmia

$ wget -O seed_list https://pastebin.com/raw/fTEg0C5T
tail -n +230 < fTEg0C5T >seed_list

$ source virtual/bin/activate
(virtual) $ scrapy crawl ahmia-tor -s DEPTH_LIMIT=1 -s
CLOSESPIDER_PAGECOUNT=2000 -s TARGET_SITES=./seed_list
(virtual) $ deactivate

```

**Figure 7 - Collection and Processing**

Perform a simple analysis using Kibana. Kibana allows a search function using a visual review of the logs, keywords, regular expressions, or defined NLP entities. Simple searches on a limited collection show a real-world email, references to hacktivists, references to exploits, a real-world ip address, and real-world names. Figure 8 displays a few techniques.

REVIEW the logs visually

The figure consists of three screenshots of the Kibana interface, demonstrating different search methods for log analysis.

**Top Screenshot:** A keyword search for `..source` is performed. The search bar shows `Search... (e.g. status:200 AND extension:PHP)`. The left sidebar shows the 'Discover' tab selected. The main panel displays two log entries. The first entry is dated October 13, 2017, at 03:24:10.000, and contains a raw text field with a URL. The second entry is dated October 13th, 2017, at 03:24:12.000, and contains a raw text field with a list of items.

**Middle Screenshot:** A keyword search for `"dot com"` is performed. The search bar shows `Search... (e.g. status:200 AND extension:PHP)`. The left sidebar shows the 'Discover' tab selected. The main panel displays one log entry dated October 13th, 2017, at 03:24:12.000, containing a raw text field with a list of items.

**Bottom Screenshot:** A regular expression search for `operationpayback` is performed. The search bar shows `Search... (e.g. status:200 AND extension:PHP)`. The left sidebar shows the 'Discover' tab selected. The main panel displays one log entry dated October 13th, 2017, at 03:24:12.000, containing a raw text field with a list of items.

Figure 8 - Analysis with Kibana

Finally, perform a link analysis using Maltego. First, start Maltego and log on using the pseudo-anonymous identity. To perform a link analysis requires a starting entity. Select Create a New Graph and open the Entity Palette. The earlier example showed a location-

to-person link analysis. This example shows a person to location link analysis. On the palette, expand personal and drag and drop a person onto the graph. Double click and rename the person entity with an actual person. Now right click on the person and run the "To URL - Using ElasticsearchMLTQuery" transform. Running the person transform will return a list of URL's that contain the person. Now right click on a URL and run the "To Location - Using Elasticsearch" transform. Running the URL transform will return a list of locations at that URL. Figure 9 displays the results.

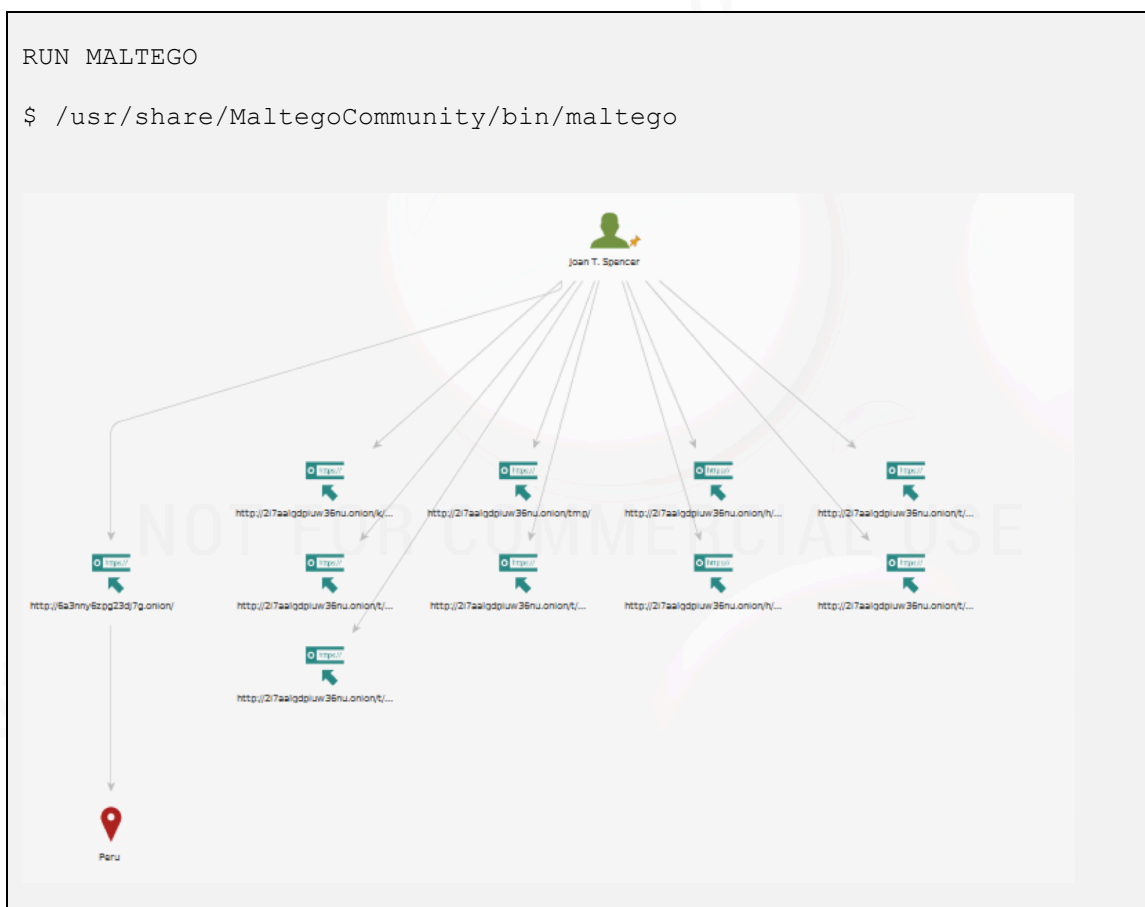


Figure 9 - Analysis with Maltego

### 3. CONCLUSION

This paper successfully explored an open-source intelligence automation toolset that scanned across the darknet. It described and shared the tools, process, and techniques to build a secure darknet connection, and then collected, processed, stored, and analyzed data. This paper showed the viability of darknet open-source intelligence using the

completed toolset. In the end, the toolset finds entities and links entities from the darknet thereby showing strong potential to aid the open source intelligence professional.

Multiple challenges arose. First, understanding and building the anonymity system amplified the desire to apply continued layering techniques for richer anonymity. However, the complexity would have increased the time, cost, and manageability. The solution was to understand that the threat profile was, in this case, minimal and keep the anonymity system simple. Next, revising the OpenNLP processor code, while minimal, was difficult but offered an opportunity to learn Java. Finally, the OpenNLP free models generated what appeared to be valid data, but the results did not prove to be actionable data due to the small run scope and limited tuning. The solution is an increased crawling and scraping run scope, tuning (such as stop-words, etc.), and training of the OpenNLP processing. In the end, however, a working toolset was the goal.

Many areas for potential growth exist. Anonymity systems and anonymity techniques are an exciting stand-alone area of research because of the highly technical nature of creating and supporting sophisticated anonymity. As it pertains to this paper, primary significance exists in revising crawling and scraping scripts to scrape closed forums by using cached identities and solving captchas (such as DeathByCaptcha, etc.) since these forums offer rich data sources. Secondary significance exists in creating transform scripts that build and display complex relationships because further analysis refines the analyst's ability in data mining. Finally, as noted earlier, value exists in large-scale testing, tuning, and training of the complete toolset for increased actionable data.



## References

- Ahmia. (2014). Ahmia search engine crawler. Retrieved from <https://github.com/ahmia/ahmia-crawler>
- Anderson, R. (1996, October). The Eternity Service. In Proceedings of PRAGOCRYPT (Vol. 96, pp. 242-252).
- Anderson, R. (2010). Security engineering: a guide to building dependable distributed systems. John Wiley & Sons.
- Bardin, J. (2012). So you Want To Be A Cyber Spook. Retrieved from <https://www.hacktivity.com/en/downloads/archives/171/&usg=AOvVaw160sW WUSEMleOFsyqcEYOU>
- Bazzell, M., & Carroll, J. (2016). The Complete Privacy & Security Desk Reference- Volume I Digital.
- Benavides, Ben (2016). Open Source Intelligence (OSINT) ToolKit On The Go. Retrieved from <http://phibetaiota.net/wp-content/uploads/2012/09/2016-OSINT-2oolKit-Benavides.pdf>
- Best, C., (2011). Challenges in open source intelligence. IEEE Intelligence and Security Informatics Conference, 58- 62.
- Beyer, M. A., & Laney, D. (2012). The importance of ‘big data’: a definition. Stamford, CT: Gartner, 2014-2018.
- Biddle, P., England, P., Peinado, M., & Willman, B. (2002, November). The darknet and the future of content protection. In ACM Workshop on Digital Rights Management (pp. 155-176). Springer, Berlin, Heidelberg.
- Butler, B., Wardman, B., & Pratt, N., (2016). "REAPER: an automated, scalable solution for mass credential harvesting and OSINT," 2016 APWG Symposium on Electronic Crime Research (eCrime), Toronto, ON, pp. 1-10.
- Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., ... & Wang, W. (2006). Data Mining Curriculum: A proposal (Version 1.0). Intensive Working Group of ACM SIGKDD Curriculum Committee, 140.
- Chaum, D. L. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM, 24(2), 84-90.

- Chaum, D. (1985). Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10), 1030-1044.
- Ciancaglini, V., Balduzzi, M., McArdle, R., & Rösler, M. (2015). The Deep Web. *Trend Micro*.
- Cimpanu C. (2017). Retrieved from <https://www.bleepingcomputer.com/news/technology/internets-largest-bitcoin-mixer-shuts-down-realizing-bitcoin-is-not-anonymous/>
- Clarke, C. (2015, August). Open Source Intelligence. An oxymoron or real intelligence? *Marine Corps Gazette*. Retrieved from <https://www.mca-marines.org/gazette/2015/08/open-source-intelligence>
- Conway, M. (2006). Terrorism and the Internet: New media—New threat? *Parliamentary Affairs*, 59(2), 283-298.
- Corona, A. (2016). Choosing a VPN. Retrieved from <https://medium.com/silk-stories/choosing-a-vpn-be-careful-here-s-why-92f2b97bb203>
- Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6), 644-654.
- Dingledine, R., & Syverson, P. (2002, March). Reliable MIX cascade networks through reputation. In *International Conference on Financial Cryptography* (pp. 253-268). Springer, Berlin, Heidelberg.
- Dingledine, R., Mathewson, N., & Syverson, P. (2004). Tor: The second-generation onion router. Naval Research Lab Washington DC.
- Edman, M., & Yener, B. (2009). On anonymity in an electronic society: A survey of anonymous communication systems. *ACM Computing Surveys (CSUR)*, 42(1), 5.
- Elasticsearch. (2010). Elasticsearch search engine based on Apache Lucene. Retrieved from <https://www.elastic.co/>
- Fachkha, C. (2015). Darknet as a Source of Cyber Threat Intelligence: Investigating Distributed and Reflection Denial of Service Attacks
- Guardian, The. (2013). Tor: 'The King of High-Secure, Low-Latency Anonymity.' *The Guardian*, October 4, World news. Retrieved from <https://www.theguardian.com/world/interactive/2013/oct/04/tor-high-secure-internet-anonymity>

- Ingersoll, G. S., Morton, T. S., & Farris, A. L. (2013). Taming text: how to find, organize, and manipulate it. Manning Publications Co.
- Interagency OPSEC Support Staff (IOSS). (1991). Compendium of OPSEC Terms. Greenbelt, Md.
- Interagency OPSEC Support Staff (IOSS), & Booz, Allen & Hamilton. (1996). Intelligence Threat Handbook. Greenbelt, Md.
- Kosala, R., & Blockeel, H. (2000). Web mining research: A survey. ACM Sigkdd Explorations Newsletter, 2(1), 1-15.
- Kibana. (2010). Kibana Elasticsearch data analytics and visualization plugin. Retrieved from <https://www.elastic.co/>
- Krizan, L. (1999). Intelligence essentials for everyone. JOINT MILITARY INTELLIGENCE COLLEGE, WASHINGTON DC.
- Layton, R., & Watters, P. A. (2015). Automating Open Source Intelligence: Algorithms for OSINT. Syngress.
- LeVeque, R. J. (2013). Top ten reasons to not share your code (and why you should anyway). SIAM News.
- Liddy, E.D. 2001. Natural Language Processing. In Encyclopedia of Library and Information Science, 2nd Ed. NY. Marcel Decker, Inc.
- Maltego. (2007). Interactive data mining using link analysis. Retrieved from <https://www.paterva.com/web7/buy/maltego-clients/maltego-ce.php>
- Mathams, R.H., (1988). The Intelligence Analyst's Notebook, Working Paper No. 151. Canberra, Strategic and Defence Studies Centre, ANU.
- McCue, C. (2014). Data mining and predictive analysis: Intelligence gathering and crime analysis. Butterworth-Heinemann.
- Mirimir, (n.d.). Privacy Guides. Including VPN's and Threat Models Guide. Retrieved from <https://www.ivpn.net/privacy-guides>
- Mitchell, R. (2015). Web scraping with Python: collecting data from the modern web. O'Reilly Media, Inc.
- Moore, D., & Rid, T. (2016). Cryptopolitik and the Darknet. Survival, 58(1), 7-38.
- Najork, M. (2009). Web crawler architecture. In Encyclopedia of Database Systems (pp. 3462-3465). Springer US.

- Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system.
- Nunes, E., Diab, A., Gunn, A., Marin, E., Mishra, V., Paliath, & Shakarian, P. (2016, September). Darknet and deepnet mining for proactive cybersecurity threat intelligence. In *Intelligence and Security Informatics (ISI)*, 2016 IEEE Conference on (pp. 7-12).
- Reelson, A. (2016). OpenNLP Ingest Processor plugin based on Apache OpenNLP. Retrieved from <https://github.com/spinscale/elasticsearch-ingest-opennlp>
- Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), 120-126.
- Robertson, J., Diab, A., Marin, E., Nunes, E., Paliath, V., Shakarian, J., & Shakarian, P. (2017). *Darkweb Cyber Threat Intelligence Mining*. Cambridge University Press.
- Rutkowska, J. (2008). The Three Approaches To Computer Security. Retrieved from <http://theinvisiblethings.blogspot.ru/2008/09/three-approaches-to-computer-security.html>
- Sanchez-Rola, I., Balzarotti, D., Santos, I. (2017). The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1251-1260.
- Schanzenbach, M. (2014, March). Hiding from Big Brother. In *Proceeding zum Seminar Future Internet (FI) und Innovative Internet Technologien und Mobilkommunikation (IITM)* (Vol. 67).
- Tabriz, P. M. (2007). Byzantine Attacks on Anonymity Systems. Doctoral dissertation, the University of Illinois at Urbana-Champaign.
- van den Brink, P., Broekman, C., Rijken, M., Oggero, S., & Verburgh, T. (2016) The Emerging Role of Social Media in Enhancing Public Security. European Union's Horizon 2020 Research and Innovation Programme.
- Ward, J. S., & Barker, A. (2013). Undefined by data: a survey of big data definitions. arXiv preprint arXiv:1309.5821.

Zhang, Y., Zeng, S., Huang, C. N., Fan, L., Yu, X., Dang, Y., ... & Chen, H. (2010, May). Developing a Dark Web collection infrastructure for computational and social sciences. In Intelligence and Security Informatics (ISI), 2010 IEEE International Conference on (pp. 59-64).

## Appendix

### Code

<https://github.com/bnafziger/ahmia-index>

<https://github.com/bnafziger/ahmia-crawler>

<https://github.com/bnafziger/elasticsearch-ingest-opennlp>

<https://github.com/bnafziger/MaltegoTransform-Python>

### Quotes

"Tor developers wanted to enlighten, but created darkness instead" (Moore, & Rid, 2016).

"The darknet-genie will not be put back into the bottle" (Biddle, England, Peinado, & Willman, 2002).

"Mathematics may be pure, but implementations embody moral values and political choices, and these choices can either advance or undermine liberty. Developers bear responsibility for their creations" (Moore, & Rid, 2016).

"Darknets are not illegal in free countries and they probably should not be. Yet these widely abused platforms ... are and should be fair game for the most aggressive intelligence and law-enforcement techniques, as well as for invasive academic research" (Moore, & Rid, 2016).

### Base Implementations

The toolset requires hardware. Elasticsearch is the primary resource consumer and drives the hardware requirements. The memory suggestion is 64 GB, but 32 GB and 16 GB are common. The CPU suggestion is a modern processor with multiple cores. Finally, the disk suggestion is SSD, but spinning media with 15K RPM drives is acceptable.

On the hardware, start building the anonymity system by installing Ubuntu. To best support anonymity, select the following screen options. First, do not install updates and do not install updates to 3rd party software until the VPN is functioning - hide all further communications behind the VPN. Second, select to encrypt the installation and select a strong security key. Finally, select a generic hostname ("ubuntu"), select a generic username ("guest"), and select a strong password. Figure 10 shows the process.

```

DOWNLOAD Ubuntu

$ wget http://releases.ubuntu.com/16.04/ubuntu-16.04-desktop-amd64.iso

CREATE USB (as seen using OSX)

$ diskutil list
$ diskutil unmountDisk /dev/disk2
$ sudo dd if=./ubuntu-16.04.2-desktop-amd64.iso of=/dev/disk2 bs=1m
$ diskutil eject /dev/disk2

INSTALL From USB

DO NOT Install Updates
DO NOT Install updates to 3rd party software

Select Erase disk and Install Ubuntu
Select Encrypt the new Ubuntu installation for Security
Select Use LVM with the new Ubuntu installation

Choose a security key
Confirm the security key

Username guest
Hostname ubuntu
Password

```

**Figure 10 - Operating System Install**

Continue building the anonymity system by establishing a new pseudo-anonymous identity. Establish a new identity by generating and tracking details: name, address, age, birth date, email address, username, password, and currency. Use fakenamgenerator.com for further ideas on details (Bardin, 2012). Now backfill a few details. Establish a new pseudo-anonymous email. Use tutanota.com or another service. Typically, this also requires a temporary email. Use incognitomail.com or another service. Create pseudo-anonymous currency. Use cash and purchase a Vanilla Visa Gift Card at a remote convenience store. While at the remote location register the card with the pseudo-anonymous identity zip code (Bazzell, & Carroll, 2016).

Continue building the anonymity system with the privacy service. Register and pay for the direct-connect VPN using the identity email, password, and currency. Install OpenVPN, download the NordVPN compressed file, extract the configuration files, and connect to the country of choice. Validate the VPN is working using curl. Figure 11 shows the process.

At this point in building the anonymity system, the only information communicated outbound on the real-world IP is one OS download, one temporary email account, one pseudo-anonymous email account, and one pseudo-anonymous VPN.

```

INSTALL OpenVPN Service

$ sudo apt-get install openvpn

$ cd /etc/openvpn

$ sudo wget -O nord.zip https://nordvpn.com/api/files/zip
$ sudo apt-get install unzip
$ sudo unzip nord.zip
$ ls -al

TEST Service

# Choose a VPN server by country code
$ sudo openvpn [file name]

# For example, Malaysia:
$ sudo openvpn my1.nordvpn.com.udpl194.ovpn

# Enter Account Credentials

$ sudo apt-get install curl

$ curl -A json 'https://ip-show.com'{"ip":"X.X.X.X","xforwarded-for-ip":null,"userAgent":"json","hostName":"","city":"Kuala Lumpur","state":"Kuala Lumpur","country":"Malaysia","zip":"50586","latitude":3.1667,"longitude":101.7,"asn":55720,"org":"Gigabit Hosting Sdn Bhd","isp":"Taman OUG Square","_comment":"This product includes GeoLite2 data created by MaxMind, available from http://www.maxmind.com."}

```

**Figure 11 - OpenVPN Install**

With the VPN functioning on the anonymity system, update and upgrade existing OS services. Figure 12 displays the process. The update and upgrade process synchronizes available packages with the source repositories and then downloads and installs any newer versions.



UPDATE AND UPGRADE

```
$ sudo apt-get update && sudo apt-get upgrade
```

**Figure 12 - OS Upgrade**

Complete building the anonymity system with the anonymity service. Install and test the TOR service. Additionally, install and configure the required Polipo cache service. Validate the TOR and Polipo services are working using curl. Figure 13 displays the process.

INSTALL TOR Service

```
$ sudo add-apt-repository "deb
http://deb.torproject.org/torproject.org $(lsb_release -cs) main"

$ gpg --keyserver keys.gnupg.net --recv
A3C4F0F979CAA22CDBA8F512EE8CBC9E886DDD89
$ gpg --export A3C4F0F979CAA22CDBA8F512EE8CBC9E886DDD89 | sudo apt-key
add -

$ sudo apt-get update
$ sudo apt-get install deb.torproject.org-keyring tor
```

INSTALL Polipo Service

```
$ sudo apt-get install polipo

$ sudo tee --append /etc/polipo/config <<-EOF
logFile=/var/log/polipo/polipo.log
disableLocalInterface=true
diskCacheRoot=""
EOF
```

TEST Services

```
$ sudo systemctl restart tor.service
$ sudo systemctl restart polipo

$ curl --proxy localhost:8123 -A json 'https://ip-show.com'
{"ip":"37.218.240.68","xforwarded-for-
ip":null,"userAgent":"json","hostName":"","city":null,"state":null,"co
untry":"Netherlands","zip":null,"latitude":52.3824,"longitude":4.8995,
"asn":133752,"org":"Leaseweb Asia Pacific pte. ltd.,"isp":"Greenhost
BV","_comment":"This product includes GeoLite2 data created by
MaxMind, available from http://www.maxmind.com."}

$ sudo systemctl stop tor.service
```

**Figure 13 - TOR & Polipo Install**

At this point in the process, building sophisticated anonymity requires additional understanding. **Understanding of adversaries, threats, threat models, and compartmentalization and isolation** See the Appendix for a few ideas on increased anonymity.

Start building the intelligence system processing service with the ingestion and storage of data using Elasticsearch. First, install the prerequisite Oracle Java JDK. Then install and test the Elasticsearch service. Finally, validate the Elasticsearch service is working using curl. Figure 14 shows the process.

```

INSTALL Oracle JDK Prerequisite

$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer

$ sudo tee --append ~/.bashrc <<-EOF
export JAVA_HOME="/usr/lib/jvm/java-8-oracle"
EOF

$ source ~/.bashrc
$ echo $JAVA_HOME

INSTALL Elasticsearch Service

$ wget
https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-
5.5.1.deb

$ shasum elasticsearch-5.5.1.deb
d6beceeb93ade6c3bc18b76a7f0e365dd95f6f52
$ sudo apt-get install ./elasticsearch-5.5.1.deb

$ sudo systemctl enable elasticsearch
$ sudo systemctl start elasticsearch

TEST Service

$ curl -X GET 'http://localhost:9200'
{
  "name" : "exelDYX",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "MwxyCXAmRuqMHcLH9VGZww",
  "version" : {
    "number" : "5.5.1",
    "build_hash" : "19c13d0",
    "build_date" : "2017-07-18T20:44:24.823Z",
    "build_snapshot" : false,
    "lucene_version" : "6.6.0"
  },

```

```
"tagline" : "You Know, for Search"
}

TUNE Service if required

/etc/Elasticsearch/jvm.xml

-Xms8g
-Xmx8g
```

**Figure 14 - Elasticsearch Install**

Continue building the intelligence system analysis service with the visual analysis of data using Kibana. Install and test the Kibana graphical interface service. Validate the Kibana service is working using curl, or by visually browsing to the service remembering that there is no data loaded yet. Figure 15 shows the process.

```
INSTALL Kibana Service

$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo
apt-key add -

$ wget https://artifacts.elastic.co/downloads/kibana/kibana-5.5.1-
amd64.deb

$ shasum kibana-5.5.1-amd64.deb
a26a909a87459afca9a93ea525228ad477ebae76
$ sudo apt-get install ./kibana-5.5.1-amd64.deb

$ sudo systemctl enable kibana.service
$ sudo systemctl start kibana.service

TEST Service

$ curl -X GET -I 'http://localhost:5601/status'
HTTP/1.1 200 OK
kbn-name: kibana
kbn-version: 5.5.1
cache-control: no-cache
content-type: text/html; charset=utf-8
content-length: 49820
accept-ranges: bytes
Date: Tue, 28 Jul 2017 02:40:08 GMT
Connection: keep-alive

BROWSE Service

http://localhost:5601/status
```

**Figure 15 - Kibana Install**

Kibana offers the ability to visually query and that ability, in and of itself, offers analytical capabilities. However, the desired outcome, outlined in the core of the paper, is to offer natural language processing (NLP) and relational link analysis. Therefore, the next step is building natural language processing (NLP) and relational link analysis.

## Anonymity

Compromising anonymity requires monitoring and or manipulating anonymity systems (Edman, & Yener, 2009). Tabriz's "Byzantine Attacks on Anonymity Systems" and Edman and Yener's "Survey of Anonymous Communication Systems," explain the classifications of adversaries. The first classification is the adversaries' capability which is passive or active and signifies the ability to monitor or manipulate traffic. Visibility is the second classification which is local or global describing the reach of the adversary. Thirdly, is mobility which is static or adaptive and designates the flexibility of the adversary. The final classification is participation which is internal or external and defines the location of the adversary within or external to the anonymity system (Tabriz, 2007; Edman, & Yener, 2009).

Maintaining anonymity requires a threat model of the adversary and the anonymity system. Mirimir's "Defining Your Threat Model" guide defines threat modeling as the process of defining what the user is protecting, what adversaries the user is protecting against, and what consequences the user might face if compromised (Mirimir, n.d.). Tabriz's "Byzantine Attacks on Anonymity Systems" succinctly outlines an analysis of common threat models using various adversaries and anonymity systems (Tabriz, 2007). However, Mirimir's guide outlines practical threat models using several anonymity systems such as layering of VPN's for privacy and TOR for anonymity, to name a few of many options (Mirimir, n.d.).

Maximizing anonymity requires compartmentalization and isolation. Compartmentalization is the act of separation of information, typically as used in the intelligence community to limit information to entities on a need to know basis. (Anderson, 2010) Isolation is the act of separation of components of systems by physical and or logical means, for instance, separation by using differing: networks, servers, virtual networks,

virtual servers, processes, and memory (Rutkowska, 2008). Together, they both play a role in a maximizing an anonymity system.

The level of compartmentalization and isolation depends on the threat level. The higher the threat, the more components that are required to protect the anonymity and privacy. With isolation, at the hardware level, there may be multiple servers and or gateways devices. At the operating system level, there may be multiple OS guest virtual machines (VM) with multiple OS VM varieties. At the network level, there may be nested VPN connections, multiple VPN providers, multiple VPN gateways, multiple TOR gateways, and connections to disparate spheres of influence, that is, disparate economic, legal, and diplomatic cooperation among countries where VPN's and TOR nodes exit. With compartmentalization, there may be multiple pseudo-anonymous identities and associated data, perhaps singular identities per OS instance and or per operation. Similar to the nesting of services, even pseudo-anonymous identities can spawn additional pseudo-anonymous identities, that is, layering of identities. Identities can require substantial effort to create and maintain, including perhaps, an understanding of foreign languages, cultures, and or locations compiled over time. The key to identity management is avoiding any contamination with a real-world identity and careful management of relationships between layered services and identities (Mimir, n.d., Bardin, 2012). See Mimir's Advanced Privacy and OPSEC Guides and Jeff Bardin's papers and talk for practical thinking and examples.

For increased anonymity, upon completion of the base VPN services steps, virtual machines offer opportunities for increased compartmentalization and isolation across multiple operating systems, multiple connections, and multiple identities. The steps below are the building blocks to multiple virtual machines. Figures 16 and 17 show the processes. Once created, each VM can exist for further layering of identities, VPN services, TOR services, and finally intelligence services.

#### INSTALL Virtual Box

```
$ sh -c "echo 'deb http://download.virtualbox.org/virtualbox/debian
'$(lsb_release -cs)' contrib' >
/etc/apt/sources.list.d/virtualbox.list"
```

```

$ wget -q https://www.virtualbox.org/download/oracle_vbox_2016.asc -O-
| sudo apt-key add -
$ wget -q https://www.virtualbox.org/download/oracle_vbox.asc -O- |
sudo apt-key add -

$ sudo apt-get update

$ wget
http://download.virtualbox.org/virtualbox/5.1.24/virtualbox-
5.1_5.1.24-117012~Ubuntu~xenial_amd64.deb

$ wget
http://download.virtualbox.org/virtualbox/5.1.24/Oracle_VM_VirtualBox_
Extension_Pack-5.1.24-117012.vbox-extpack

$ sudo apt-get install ./virtualbox-5.1_5.1.24-
117012~Ubuntu~xenial_amd64.deb dkms

$ sudo VBoxManage extpack install
./Oracle_VM_VirtualBox_Extension_Pack-5.1.24-117012.vbox-extpack

```

**Figure 16 - Virtual Machine Build**

```

DOWNLOAD

$ wget http://releases.ubuntu.com/16.04/ubuntu-16.04-desktop-amd64.iso

INSTALL Virtual Workstation

$ export VM='Ubuntu-16.04-Desktop-1'
$ VBoxManage createhd --filename $VM.vdi --size 32768
$ VBoxManage createvm --name $VM --ostype "Ubuntu_64" --register

$ VBoxManage storagectl $VM --name "SATA Controller" --add sata --
controller IntelAHCI
$ VBoxManage storageattach $VM --storagectl "SATA Controller" --port 0
--device 0 --type hdd --medium $VM.vdi
$ VBoxManage storagectl $VM --name "IDE Controller" --add ide
$ VBoxManage storageattach $VM --storagectl "IDE Controller" --port 0
--device 0 --type dvddrive --medium /home/guest/ubuntu-16.04.2-
desktop-amd64.iso

$ VBoxManage modifyvm $VM --ioapic on
$ VBoxManage modifyvm $VM --boot1 dvd --boot2 disk --boot3 none --
boot4 none
$ VBoxManage modifyvm $VM --memory 1024 --vram 128
$ VBoxManage modifyvm $VM --nic1 bridged --nictype1 82540EM --
bridgeadapter1 eno1

```

**Figure 17 - Virtual Guest Build**

For additional anonymity, accomplishing disparate spheres of influence is a matter of selecting proper countries when connecting via VPN's and TOR exit nodes. In this case, choosing multiple non "fourteen eyes" countries, where "fourteen eyes" countries are countries with cooperating intelligence services (Corona, 2016). Figure 18 shows the process.

```
CONNECT VPN

$ sudo openvpn [file name using country code]

# For example, Malaysia:
$ sudo openvpn my1.nordvpn.com.udp1194.ovpn

# validate VPN country
$ curl -A json 'https://ip-show.com'

CONNECT TOR

# establish TOR control password
$ tor --hash-password "MiningInTheDark"
16:E43ACCDB29922EEB60D7E4C664ECAF390B1836628256662243C6240420

# configure TOR control data and
# configure TOR exit node countries using country codes
$ vi /etc/tor/torrc
ControlPort 9051
HashedControlPassword
16:E43ACCDB29922EEB60D7E4C664ECAF390B1836628256662243C6240420
CookieAuthentication 1
ExitNodes {fi},{ie},{md},{bz},{hz},{sg},{sc},{ro},{pa},{cy},{fi},{ba}

$ sudo systemctl restart tor.service
$ sudo systemctl restart polipo

# validate TOR exit node country
$ curl --proxy localhost:8123 -A json 'https://ip-show.com'

# controlling circuits and viewing circuits using TOR controls
$ echo -e 'AUTHENTICATE "MiningInTheDark"\r\nsignal NEWNYM\r\nQUIT' | nc 127.0.0.1 9051
$ echo -e 'AUTHENTICATE "MiningInTheDark"\r\ngetinfo stream-status\r\nQUIT' | nc 127.0.0.1 9051
$ echo -e 'AUTHENTICATE "MiningInTheDark"\r\ngetinfo circuit-status\r\nQUIT' | nc 127.0.0.1 9051
```

**Figure 18 - Disparate Spheres of Influence Configuration**

## OpenNLP Document Categorizer

The Elasticsearch OpenNLP Ingest Processor code revisions now hold DOCCAT model code. To re-build the OpenNLP Ingest Processor with DOCCAT code, download the OpenNLP tools with the doccattrainer binary. Use the doccattrainer binary to model the generic category word list or training data set. Detection accuracy requires a larger tailored training data set than seen in this instance. Finally, copy the resulting DOCCAT model into the processor resources location and compile the processor. As previously completed, install and configure the processor with the DOCCAT model reference, and restart Elasticsearch. Figure 19 displays the process.

```
BUILD OpenNLP Ingest Processor Model for DOCCAT

$ cd /home/guest/Desktop/code/

$ wget -O opennlp-tools-1.5.0-bin.tar.gz
https://downloads.sourceforge.net/project/opennlp/OpenNLP%20Tools/1.5.0/opennlp-tools-1.5.0-bin.tar.gz?r=https%3A%2F%2Fsourceforge.net%2Fprojects%2Fopennlp%2F&ts=1503413391&use_mirror=ayera

$ tar xvfz opennlp-tools-1.5.0-bin.tar.gz

$ cd /home/guest/Desktop/code/opennlp-tools-1.5.0/bin

$ wget -O en-doccat-category.tsv
https://gist.githubusercontent.com/mbejda/184d3a589caa50e7a43d/raw/11a5472cbea6b9c1ce31e1ab4b0995dlee80765/hashtagCategories.tsv

$ ./opennlp DoccatTrainer -model en-doccat-category.bin -lang en -data
en-doccat-category.tsv -encoding UTF-8

Indexing events with TwoPass using cutoff of 5
Computing event counts... done. 274 events
Indexing... Dropped event washing_machine:[bow=Joint, bow=families]
....
Writing document categorizer model ... done (0.232s)

Wrote document categorizer model to
path: /home/guest/Desktop/code/apache-opennlp-1.5.0/bin/en-doccat-
category.bin

Execution time: 1.771 seconds

$ cp /home/guest/Desktop/code/apache-opennlp-1.5.0/bin/en-doccat-
category.bin /home/guest/Desktop/code/elasticsearch-ingest-
opennlp/src/test/resources/models/en-doccat-category.bin
```



COMPILE

```
$ cd /home/guest/Desktop/code/elasticsearch-ingest-opennlp/
$ gradle clean check
$ gradle assemble
```

INSTALL

```
$ cd /usr/share/elasticsearch/

$ sudo bin/elasticsearch-plugin remove ingest-opennlp --purge

$ sudo bin/elasticsearch-plugin install
file:///home/guest/Desktop/code/elasticsearch-ingest-opennlp/build/distributions/ingest-opennlp-5.5.1.1-SNAPSHOT.zip

$ sudo tee --append /etc/elasticsearch/elasticsearch.yml <<-EOF
ingest.opennlp.model.file.category: en-doccat-category.bin
EOF

$ sudo systemctl restart elasticsearch

$ sudo tail /var/log/elasticsearch/elasticsearch.log

[2017-08-21T21:35:16,787][INFO ][o.e.p.PluginsService ] [WCCCQsy]
loaded plugin [ingest-opennlp]
[2017-08-21T21:35:22,818][INFO ][d.s.e.i.o.OpenNlpService ] [WCCCQsy]
Read models in [522.2micros] for [category, dates, locations, persons]
```

**Figure 19 - OpenNLP with DOCCAT Build**

Once the Elasticsearch OpenNLP Processor is functioning, confirm it is properly processing data by using curl or by visually browsing to the service. Create a sample database with the OpenNLP pipeline, and then insert data. The queried data show the detected NLP DOCCAT category entity from the text, albeit a generic category, in this instance, due to the smaller generic training data set size. To visually query, once again, requires configuring an index in Kibana. Figure 20 displays the process.

TEST Elasticsearch OpenNLP Ingest Processor

```
# wipe database
curl -X DELETE "localhost:9200/my_index/"
curl -X GET 'http://localhost:9200/my_index/_search?q=*&pretty'

# wipe pipeline
$ curl -X DELETE "localhost:9200/_ingest/pipeline/opennlp-pipeline/"
$ curl -X GET localhost:9200/_ingest/pipeline/opennlp-pipeline

# create database
$ curl -X PUT -i "localhost:9200/my_index/" -d '
```

```
{
  "mappings": {
    "my_type": {
      "properties": {
        "my_field": { "type": "string" }
      }
    }
  }
}

# create pipeline
$ curl -X PUT localhost:9200/_ingest/pipeline/opennlp-pipeline -d ' {
"description": "A pipeline to do named entity extraction",
"processors": [ { "opennlp" : { "field" : "my_field" "ignore_missing":
true } } ] } '

# add data
$ curl -X PUT 'localhost:9200/my-index/my-type/1?pipeline=opennlp-
pipeline' -d ' { "my_field" : "Kobe Bryant was one of the best
basketball players of all times. Not even Michael Jordan has ever
scored 81 points in one game. Munich is really an awesome city, but
New York is as well. Yesterday has been the hottest day of the year."
} '

# query data
$ curl -X GET 'localhost:9200/my-index/my-type/1'
{"_index":"my-index","_type":"my-
type","_id":"1","_version":4,"found":true,"_source":{"my_field":"Kobe
Bryant was one of the best basketball players of all times. Not even
Michael Jordan has ever scored 81 points in one game. Munich is really
an awesome city, but New York is as well. Yesterday has been the
hottest day of the year.,"entities":{"persons":["Kobe
Bryant","Michael
Jordan"],"dates":["Yesterday"],"locations":["Munich","New
York"],"category":["Bike"]}}}
```



The screenshot shows the Kibana web interface. On the left is a sidebar with navigation links: Discover, Visualize, Dashboard, and Timeline. The main area displays a search result for a document with the ID '1'. The document's source is a text snippet about Kobe Bryant and Michael Jordan. The 'entities' field contains a list of named entities: 'persons' (Kobe Bryant, Michael Jordan), 'dates' (Yesterday), 'locations' (Munich, New York), and 'category' (Bike). The interface also shows a search bar at the top and a 'Discover' button on the left.

Figure 20 - OpenNLP with DOCCAT Testing (Reelson, 2016)