Welcome to

3. Storing and Processing data

KEA Kompetence SIEM and Log Analysis

Henrik Kramselund he/him han/ham hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Github

3-storing-and-processing.tex in the repo security-courses

Goals for today

Todays goals:

- Talk about example SIEM components
- Realize Elasticsearch is a very common *storage system* for infosec products
- Play with Elasticsearch, also share experiences with running it in production
- Get your data into Elasticsearch! Your Zeek data!

Photo by Thomas Galler on unsplash

Plan for today

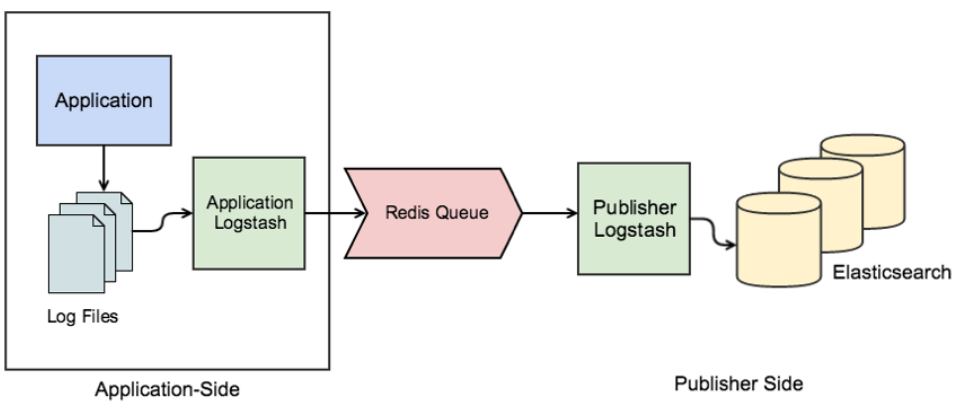
Subjects

- Elasticsearch – a highly efficient data store
- Elasticsearch SIEM
- Along the way, operations for Elasticsearch and SIEM architectures

Exercise theme: Storing and processing

- Running Elasticsearch
- What can you put into Elasticsearch

Today we need Elastic stack running!



Note: Kibana makes it easy to use sample data, feel free to experiment!

Elasticsearch and Kibana are *services* which open a listening socket/port. zSo access ES via https://127.0.0.1:9200 and Kibana via https://127.0.0.1:5601 on your Debian using a browser or Postman

Reading Summary

CIP 4 A Data-Centric Approach to Security Monitoring

Skim read: CIP 7 Tools of the Trade, need to know NetFlow, DNS, and HTTP proxy logs in the real-world

Skim read: DDS 8. Breaking Up with Your Relational Database

Reading Summary, continued

> You could **buy a bunch of expensive gear**, point it all to a **log management** or a **security incident and event management (SIEM)** system, and let it automatically tell you what it knows. Some incident response teams may start this way, but unfortunately, many never evolve. **Working only with what the SIEM tells you**, versus what you have configured it to tell you **based on your contextual data**, will invariably fail. Truly demonstrating **value** from security monitoring and incident response **requires a major effort**. As with all projects, **planning** is the most important phase. **Designing** an approach that works best for you requires significant effort up front, but offers a great payout later in the form of **meaningful incident response and overall improved security**.

Source: CIP 4 A Data-Centric Approach to Security Monitoring (bold by me)

- I recommend pre-filtering, see what noise your devices *would send*
  "Collecting only relevant data can have a direct impact on reducing costs as well."
- Same is recommended in CIP page 50: Just the Facts
- Normalization – "Data normalization for the purposes of log mining is the process by which a portion, or field, of a log event is transformed into its canonical form."

Metadata – enrichment

Metadata	Metacategory: Data
The DNS lookup occurred at a certain time.	Timestamp: 278621182
The internal host sent a DNS PTR request.	Network protocol: DNS PTR
The internal host had a hostname.	Location: Desktop subnet
	Source IP Address: 1.1.1.2
	Hostname: windowspc22.company.com
The internal host resolved an external host.	Location: External
	Destination IP Address: 255.123.215.3
	Hostname: dgf7adfnkjhh.com
The external host was hosted by a dynamic DNS provider.	Network: Shady DDnS Provider Inc.
	ASN: SHADY232
	Reputation: Historically risky network
The remote hostname appeared randomly generated.	Hostname: dgf7adfnkjhh.com
	Category: Unusual, nonlinguistic

Source: picture from Crafting the InfoSec Playbook, CIP

Metadata + Context

Reading Summary, continued

Skim read: CIP 7 Tools of the Trade, need to know NetFlow, DNS, and HTTP proxy logs in the real-world

- Defense in Depth – we will never catch everything
- Log Management: The Security Event Data Warehouse
- Intrusion Detection Isn't Dead
- DNS, the One True King – Logging and analyzing DNS transactions, Blocking DNS requests or responses
- HTTP Is the Platform: Web Proxies – Web proxies allow you to solve additional security problems
- Rolling Packet Capture – In a perfect world, we would have full packet capture everywhere

Reading Summary, Intrusion Kill Chains

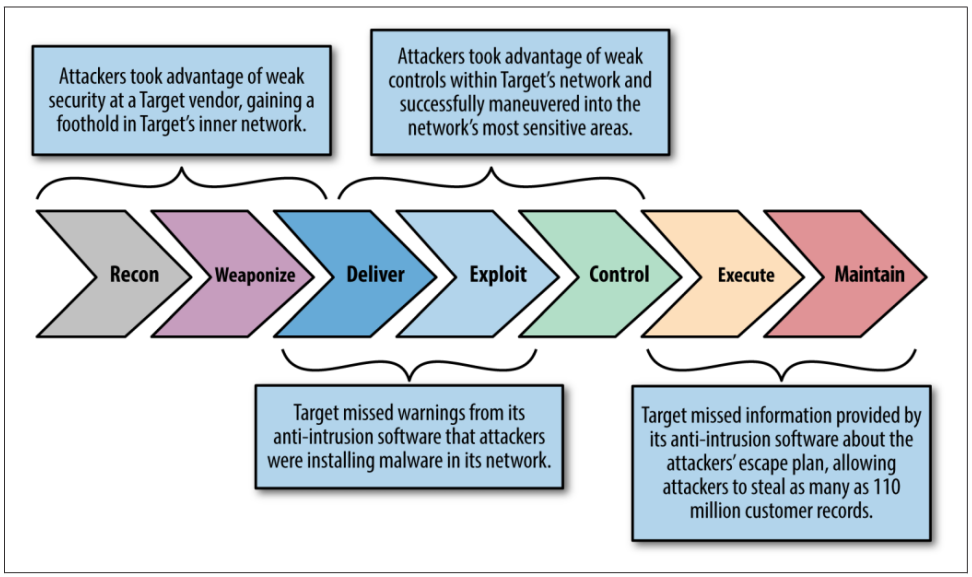


Figure 7-1. The kill chain

- See also *Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains*, Eric M. Hutchins , Michael J. Cloppert, Rohan M. Amin, Ph.D. Lockheed Martin Corporation

  https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf

Reading Summary, continued

> Relational databases (RDBMS) have been around since the 1970s when Edgar Codd proposed "a relational model of data for large shared data banks" as an alternative to the network models—heavily inter-linked, on-disk structures—prevalent at that time.

Source: DDS 8. Breaking Up with Your Relational Database

- A Primer on SQL/RDMBS Databases – read if you don't know about relational databases
- Constrained by Schema
- Exploring Alternative Data Stores – BerkeleyDB, MongoDB
- Redis is an open source, BSD licensed, advanced key-value store (http://redis.io/) – great buffer between systems
- Apache Hadoop https://hadoop.apache.org/ map and reduce, big data tools
- Perhaps checkout SQLite https://www.sqlite.org/

Example data store: Elasticsearch

> Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Elasticsearch is developed in Java.

Source: Wikipedia https://en.wikipedia.org/wiki/Elasticsearch

- Initial release 8 February 2010
- Open core means parts of the software are licensed under various open-source licenses (mostly the Apache License)
- Various browser tools and plugins for ES exist, to make life easier
- I often use ES for storing Log Messages and Events from multiple systems, a SIEM Security information and event management.

Elasticsearch

> **ElasticSearch consumes practically anything you give it** and provides straightforward ways to ask it questions and get data out of it. You just need to feed it **semi- or unstructured data** and fold in some domain intelligence to enable smart indexing. It works its multi-node NoSQL magic in conjunction with **a layer of full-text searching** to give you **almost instantaneous query results even for large amounts of data.**
> Source: DDS 8. Breaking Up with Your Relational Database

- Elasticsearch SIEM – from Elastic
- Wazuh – agent for clients, log events, integrity protection etc.
- HELK – all-in one hunting system
- ElastiFlow – netflow system
- Arkime (renamed recently from Moloch) – packet capture

Lots of commercial systems, and lots of companies providing cloud logging platform

Microsoft Azure promotes Sentinel – cloud based SIEM

https://azure.microsoft.com/da-dk/services/azure-sentinel/

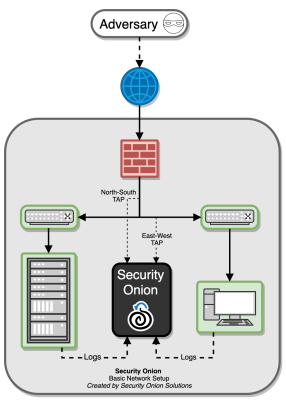
Arkime

Full Packet Capture
Arkime (formerly Moloch) is a large scale, open source, indexed packet capture and search tool.
This project has experienced significant growth, adoption, and change over the last eight years. We are now at a new milestone and believe it's the right time to rename our project to Arkime!
... On basic commodity hardware, it is easy to get 3Gbps or more, depending on the number of CPUs available to Arkime and what else the machine is doing.

Source: Picture and text from https://arkime.com/

- I haven't tried it in real life
- Note also recommendation for Network Packet Broker, example Arista - https://www.arista.com/en/solutions/tap-aggregation

Architecture for packet capture



Source: picture from https://docs.securityonion.net/en/2.3/introduction.html

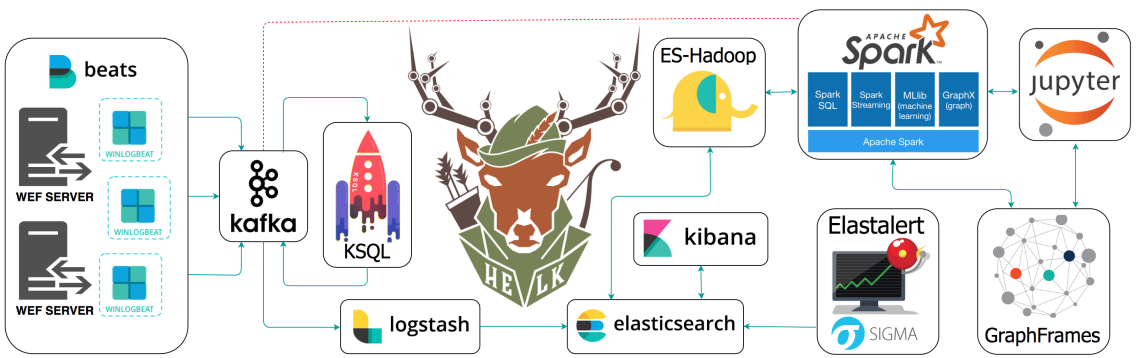
- Note the terminology North-South – from the internet into the systems
- East-West – horizontal traffic inside the data center
- See also from Security Onion https://docs.securityonion.net/en/2.3/architecture.html#architecture

ElastiFlow

ElastiFlow™ provides network flow data collection and visualization using the Elastic Stack (Elasticsearch, Logstash and Kibana). It supports Netflow v5/v9, sFlow and IPFIX flow types (1.x versions support only Netflow v5/v9).

Source: Picture and text from https://github.com/robcowart/elastiflow
PS I havent tried it in real life, yet

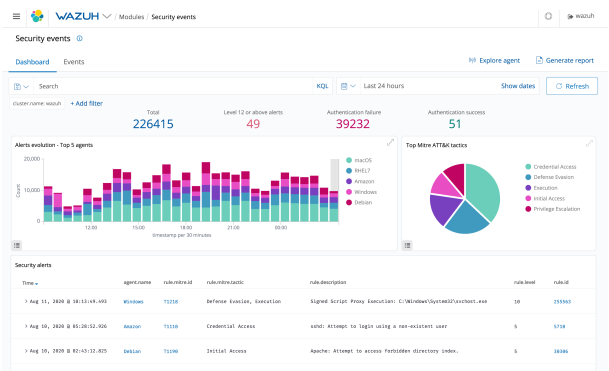
Example architecture: The Hunting ELK



The Hunting ELK or simply the HELK is one of the first open source hunt platforms with advanced analytics capabilities such as SQL declarative language, graphing, structured streaming, and even machine learning via Jupyter notebooks and Apache Spark over an ELK stack. This project was developed primarily for research, but due to its flexible design and core components, it can be deployed in larger environments with the right configurations and scalable infrastructure.
Source: text and picture from https://thehelk.com/intro.html

- You might consider this an example architecture for a SIEM, lot of components

Example system: Wazuh



Wazuh agents scan the monitored systems looking for malware, rootkits and suspicious anomalies. They can detect hidden files, cloaked processes or unregistered network listeners, as well as inconsistencies in system call responses.
Source: text and picture from https://wazuh.com/

- Wazuh initially a fork of the OSSEC project, and has integration with Elastic Stack

Wazuh agent

The Wazuh lightweight agent is designed to perform a number of tasks with the objective of detecting threats and, when necessary, trigger automatic responses. The agent core capabilities are:

The Wazuh agents run on many different platforms, including Windows, Linux, Mac OS X, AIX, Solaris and HP-UX. They can be configured and managed from the Wazuh server.

Source: https://wazuh.com/

- Log and events data collection
- File and registry keys integrity monitoring
- Inventory of running processes and installed applications
- Monitoring of open ports and network configuration
- Detection of rootkits or malware artifacts
- Configuration assessment and policy monitoring
- Execution of active responses

Chapter 9: Service API and Contract Design with REST Services and Microservices

From the KEA course system integration:

> REST service contracts are typically designed around the primary functions of HTTP methods, which make the documentation and expression of REST service contracts distinctly different from operation-based Web service contracts. Regardless of the differences in notation, the same overarching contract-first approach to designing REST service contracts is paramount when building services for a standardized service inventory.

- REST entity service contracts are typically dominated by service capabilities that include inherently idempotent and reliable GET, PUT, or DELETE methods
- This chapter provides service contract design guidance for service candidates modeled as a result of the service-oriented analysis stage covered in Chapter 7.

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

REST Service

Figure 9.1
An entity service based on the Invoice business entity that defines a functional scope that limits the service capabilities to performing invoice-related processing. This agnostic Invoice service will be reused and composed by other services within the same service inventory in support of different automated business processes that need to process invoice-related data. This particular invoice service contract displays two service capabilities based on primitive methods and two service capabilities based on complex methods.

- Very typical REST URL/method `GET /invoice/{invoice-id}`

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

REST service contracts

The following is a series of common guidelines and considerations for designing REST service contracts.

- Uniform Contract Design Considerations
- Designing and Standardizing Methods
- Designing and Standardizing HTTP Headers
- Designing and Standardizing HTTP Response Codes
- Customizing Response Codes
- Designing Media Types

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

Designing and Standardizing HTTP Response Codes

- 100-199 are informational codes used as low-level signaling mechanisms, such as a confirmation of a request to change protocols
- 200-299 are general success codes used to describe various kinds of success conditions
- 300-399 are redirection codes used to request that the consumer retry a request to a different resource identifier, or via a different intermediary
- 400-499 represent consumer-side error codes that indicate that the consumer has produced a request that is invalid for some reason, example 404 file not found
- 500-599 represent service-side error codes that indicate that the consumer's request may have been valid but that the service has been unable to process it for internal reasElasticsearch exposes REST APIs that are used by the UI components and can be called directly to configure and access Elasticsearch features.ons.

Source:

Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017

Elasticsearch REST

```
curl -X GET "localhost:9200/_cluster/health?wait_for_status=yellow&timeout=50s&pretty"
```

- Elasticsearch exposes REST APIs that are used by the UI components and can be called directly to configure and access Elasticsearch features.
- https://www.elastic.co/guide/en/elasticsearch/reference/current/rest-apis.html
- So REST is used for putting data, using `PUT` and `POST`
- And REST is used for getting data with `GET`, but also getting information about the Elasticsearch system itself, cluster health etc.
- It supports advanced querying through the API and parallel execution of searches across a cluster of nodes

Elasticsearch tutorials

Elasticsearch is a distributed document store. Instead of storing information as rows of columnar data, Elasticsearch stores complex data structures that have been serialized as JSON documents. When you have multiple Elasticsearch nodes in a cluster, stored documents are distributed across the cluster and can be accessed immediately from any node.

When a document is stored, it is indexed and fully searchable in near real-time–within 1 second. Elasticsearch uses a data structure called an inverted index that supports very fast full-text searches. An inverted index lists every unique word that appears in any document and identifies all of the documents each word occurs in.

...

The Elasticsearch REST APIs support structured queries, full text queries, and complex queries that combine the two. Structured queries are similar to the types of queries you can construct in SQL.

- https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html
- Elasticsearch Tutorials - Officlal Guide https://www.elastic.co/elasticsearch/tutorials

Elasticsearch – Analyzing your data

Elasticsearch aggregations enable you to build complex summaries of your data and gain insight into key metrics, patterns, and trends. Instead of just finding the proverbial “needle in a haystack”, aggregations enable you to answer questions like:

- How many needles are in the haystack?
- What is the average length of the needles?
- What is the median length of the needles, broken down by manufacturer?
- How many needles were added to the haystack in each of the last six months?

Source: https://www.elastic.co/guide/en/elasticsearch/reference/current/search-analyze.html

Searching in Elasticsearch

```
GET /my-index-000001/_search
```

```
{
  "query": {
    "match": {
      "user.id": "kimchy"
    }
  }
}
```

- Can use cURL, Python, Postman or any other REST client
- Indices are one of the main terms

Response


```
{
  "took": 5,
  "timed_out": false,
  "_shards": {
    "total": 1,
    "successful": 1,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 1,
      "relation": "eq"
    },
    "max_score": 1.3862942,
    "hits": [
          {
            "_index": "my-index-000001",
            "_type": "_doc",
            "_id": "kxWFcnMByiguvud1Z8vC",
            "_score": 1.3862942,
```

```
      "_source": {
        "@timestamp": "2099-11-15T14:12:12",
        "http": {
          "request": {
            "method": "get"
          },
          "response": {
            "bytes": 1070000,
            "status_code": 200
          },
          "version": "1.1"
        },
        "message": "GET /search HTTP/1.1 200 1070000",
        "source": {
          "ip": "127.0.0.1"
        },
        "user": {
          "id": "kimchy"
        }
      }
    }
  ]
}
```


ElasticSearch Index

Basic Definition

An index is defined as:

An index is like a ‘database’ in a relational database. It has a mapping which defines multiple types. An index is a logical namespace which maps to one or more primary shards and can have zero or more replica shards.

The easiest and most familiar layout clones what you would expect from a relational database. You can (very roughly) think of an index like a database.

`MySQL => Databases => Tables => Columns/Rows`

`Elasticsearch => Indices => Types => Documents with Properties`

Source: https://www.elastic.co/blog/what-is-an-elasticsearch-index

Indices for logging

Indices for Logging

Now, the reality is that Indices/Types are much more flexible than the Database/Table abstractions we are used to in RDBMs. They can be considered convenient data organization mechanisms, with added performance benefits depending on how you set up your data.

To demonstrate a radically different approach, a lot of people use Elasticsearch for logging. A standard format is to assign a new index for each day. Your list of indices may look like this:

- logs-2013-02-22
- logs-2013-02-21
- logs-2013-02-20

Elasticsearch allows you to query multiple indices at the same time, so it isn't a problem to do:

```
<strong>$ curl -XGET localhost:9200/logs-2013-02-22,logs-2013-02-21/Errors/_search?
query="q:Error Message"
</strong>
```

Source: https://www.elastic.co/blog/what-is-an-elasticsearch-index

Elasticsearch SIEM

Elastic Common Schema (ECS)

The Elastic Common Schema (ECS) defines a common set of fields for ingesting data into Elasticsearch. A common schema helps you correlate data from sources like logs and metrics or IT operations analytics and security analytics.

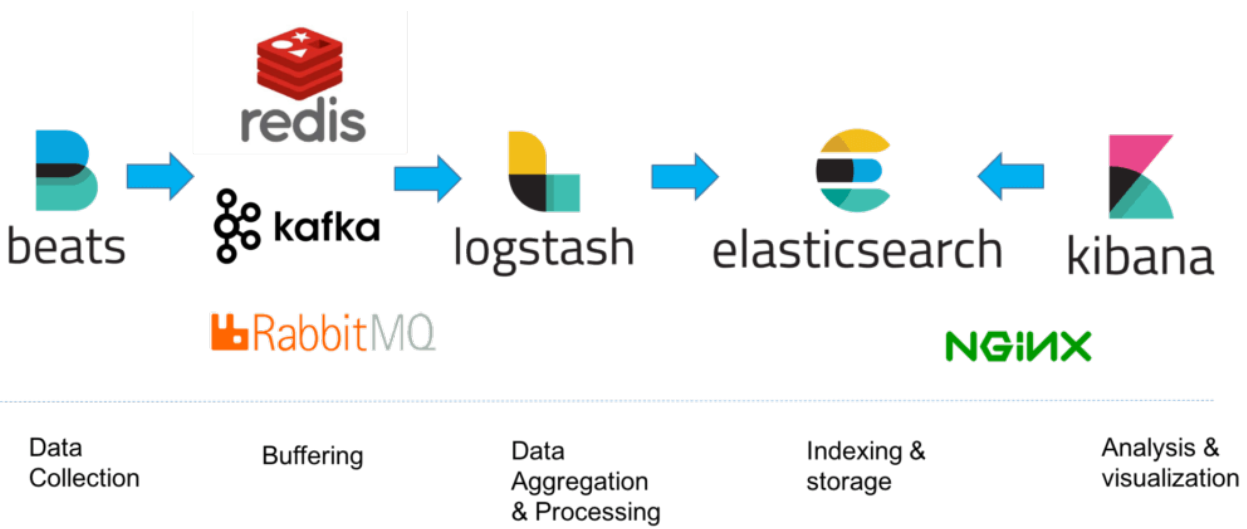
- Some structure is useful, Elastic Common Schema (ECS)
  https://github.com/elastic/ecs
- I would use their schemas for a green field deployment,
  as they have been expanded and developed over some time
- Correlation becomes implicit in every search!
  – hint/fact from https://www.elastic.co/webinars/introducing-the-elastic-common-schema

Common Event Format (CEF)

The format called Common Event Format (CEF) can be readily adopted by vendors of both security and non-security devices. This format contains the most relevant event information, making it easy for event consumers to parse and use them. To simplify integration, the syslog message format is used as a transport mechanism. This applies a common prefix to each message, containing the date and hostname, as shown below.

- Common Event Format (CEF) originally from ArcSight
- Trying to move vendors from unstructure syslog messages
- https://www.elastic.co/guide/en/beats/filebeat/7.10/filebeat-module-cef.html

Architecture



- Real production environments often add some buffering in between
- Allows the ingestion to become more smooth, no lost messages

Buy or DIY?

> DNSDB is a database that stores and indexes both the passive DNS data available via Farsight Security's Security Information Exchange as well as the authoritative DNS data that various zone operators make available.

Source: from https://docs.dnsdb.info/

- Excellent services can be bought, have used https://team-cymru.com/
- Compare using https://docs.dnsdb.info/ Farsight DNSDB API documentation
- https://nullsecure.org/building-your-own-passivedns-feed/
- Lots of examples for adding functionality, building and expanding SIEM and log systems
- I usually go to Github and have found a lot of useful tools

Team Cymru

We operate as our own ISP and are part of the fabric of the internet. We've amassed an unmatched number of data sharing partnerships with operators worldwide, in addition to gathering threat intelligence from a global grid of sensors, honeypots, darknets and crawlers. We give you our visibility via our Pure Signal™ platform, Augury™.

- Trace threat actors through dozens of proxies and VPNs.
- Map the extended infrastructure.
- Preemptively block associated IPs.
- Then monitor these threats to defend against them indefinitely.

Source: from https://team-cymru.com/

- Often you need sources that are hard to get
- Many vendors integrate sources into other products too
- Firewalls and Load balancing products that include reputation lists

The Spamhaus Don't Route Or Peer Lists

> The Spamhaus Don't Route Or Peer Lists
> DROP (Don't Route Or Peer) and EDROP are advisory "drop all traffic"lists, consisting of stolen 'hijacked' netblocks and netblocks controlled entirely by criminals and professional spammers. DROP and EDROP are a tiny subset of the SBL designed for use by firewalls and routing equipment.

http://www.spamhaus.org/drop/

- When your SIEM alerts you, you need tools to block and restrict
- Recommend adding empty blocking access control lists etc. to your network infrastructure
- Add premade blocking to your name servers, proxy servers, recursive servers
- Recommend implementing country lists

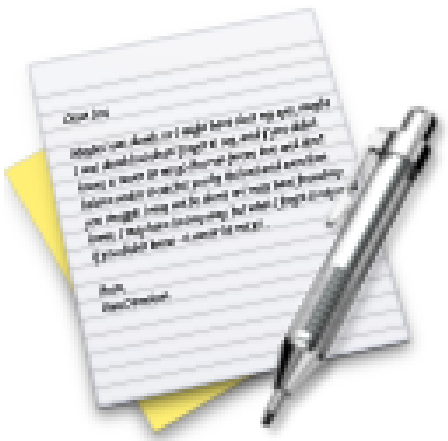
Exercise

Now lets do the exercise

⚠ Getting started with the Elastic Stack – 60min

which is number **7** in the exercise PDF.

Exercise



Now lets do the exercise

⚠ Use Ansible to install programs – 10-60min

which is number **8** in the exercise PDF.

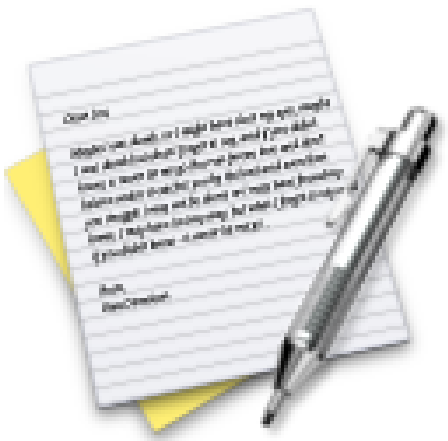
Exercise

Now lets do the exercise

⚠ Postman API Client – 20min

which is number **5** in the exercise PDF.

Exercise

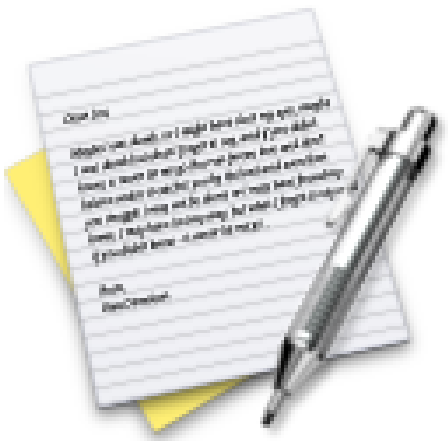


Now lets do the exercise

ℹ Making requests to Elasticsearch – 15-75min

which is number 12 in the exercise PDF.

Exercise

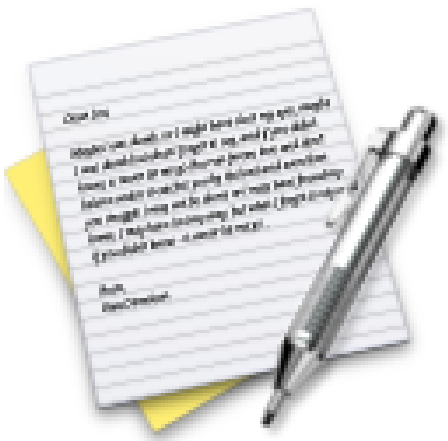


Now lets do the exercise

⚠ Find Indices in Elasticsearch – 15min

which is number **27** in the exercise PDF.

Exercise



Now lets do the exercise

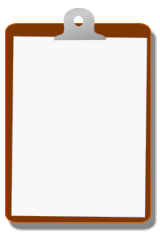
⚠ Zeek Data in Elasticsearch – 30min

which is number **28** in the exercise PDF.

Alerting

- Alerting is included in Elasticsearch, and numerous other tools
- Typically we need some destination for the alerts.
- Email alerts require some endpoint SMTP server
- Chat programs like Slack, IRC, HipChat and others
- Dashboard with colors can show alerts

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools