

KRYCY LAB 1C

**Paweł Popiołek, Michał Wawrzyńczak, Mateusz Borkowski,
Paweł Gryka**

Testowanie dostępnych silników reguł

Zdecydowaliśmy się przetestować silnik ElastAlert, który jest częścią środowiska HELK.

Zdecydowaliśmy się postawić środowisko HELK lokalnie na maszynie wirtualnej.

Zdecydowaliśmy się zaimportować dane podane jako przykładowe w tutorialu na

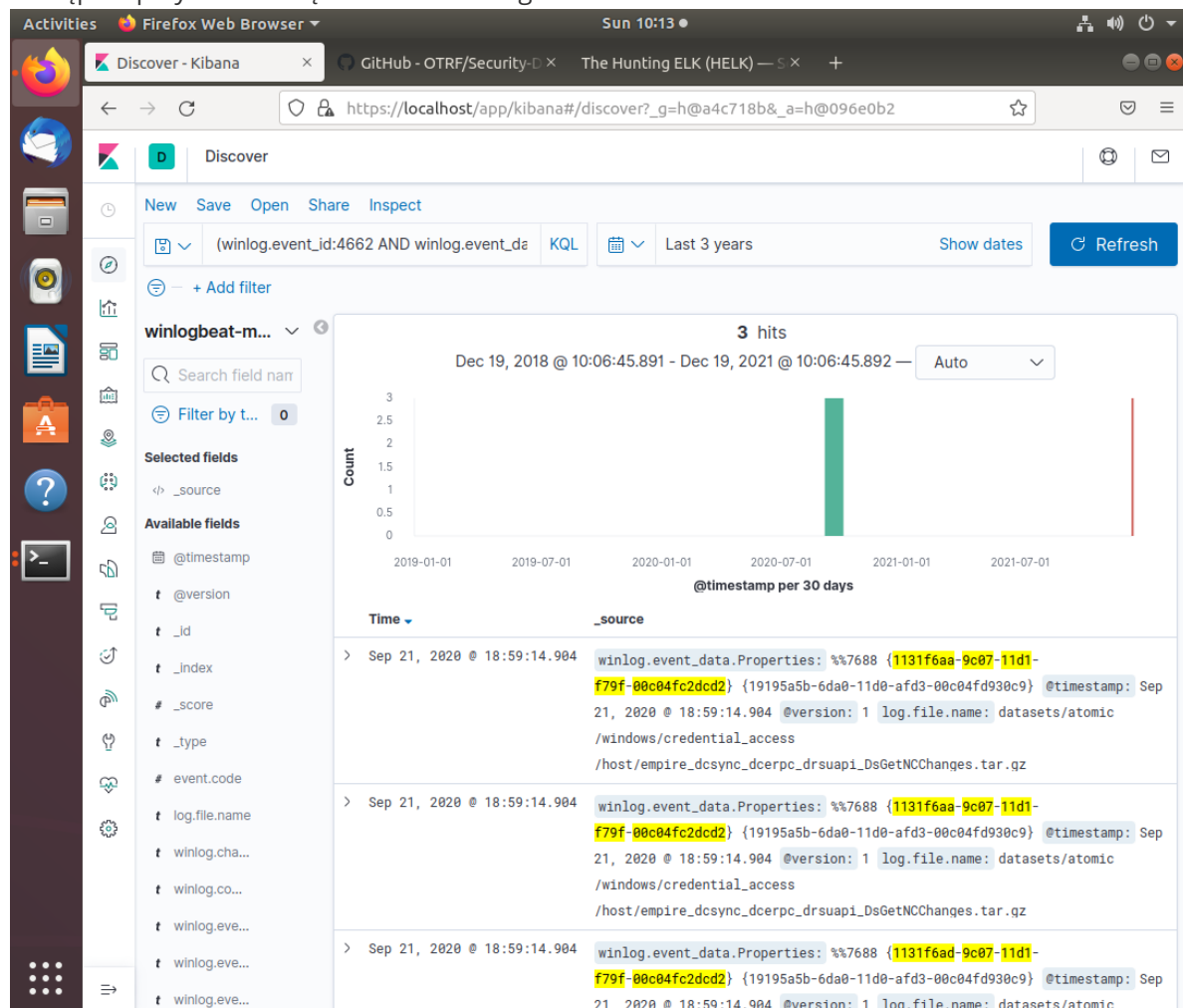
<https://securitydatasets.com/consume/helk.html>

Import danych

```
((Security-Datasets) ) ubuntu@ubuntu1804:~/Security-Datasets$ curl -XDELETE localhost:9200/winlogbeat-mordor
{"acknowledged":true}((Security-Datasets) ) ubuntu@ubuntu1804:~/Security-Datasets$
((Security-Datasets) ) ubuntu@ubuntu1804:~/Security-Datasets$ python ~/Security-Datasets/scripts/data-shippers/M
ordor-Elastic.py --url http://localhost:9200 inputs datasets/atomic/windows/credential_access/host/empire_dcsync
_dcerpc_drсуapi_DsGetNCChanges.tar.gz
Initializing Elasticsearch connection and index...
Calculating total file size...
100% (1 of 1) |#####| Elapsed Time: 0:00:00 Time: 0:00:00
Importing dataset datasets/atomic/windows/credential_access/host/empire_dcsync_dcerpc_drсуapi_DsGetNCChanges.tar
.gz
- Importing member file empire_dcsync_dcerpc_drсуapi_DsGetNCChanges_2020-09-21185829.json...
- Imported 8465 events, 0 failed
100% of 37.0 MiB |#####| Elapsed Time: 0:00:05 Time: 0:00:05
Imported 8465 log records, 0 failed.
((Security-Datasets) ) ubuntu@ubuntu1804:~/Security-Datasets$
```

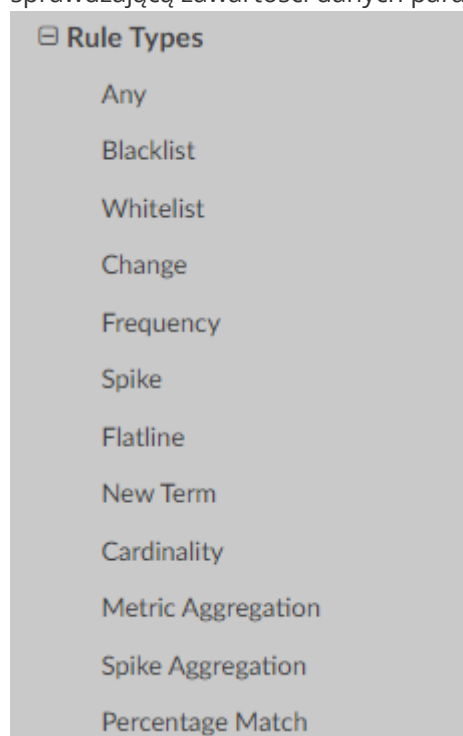
Kibana

Sprawdziliśmy czy dane zostały poprawnie zaimportowane do Elasticsearch'a i możemy przeglądać je w Kibane. Dodatkowo sprawdziliśmy z jakiego okresu pochodzą te dane co następnie przyda nam się w testowaniu reguł.

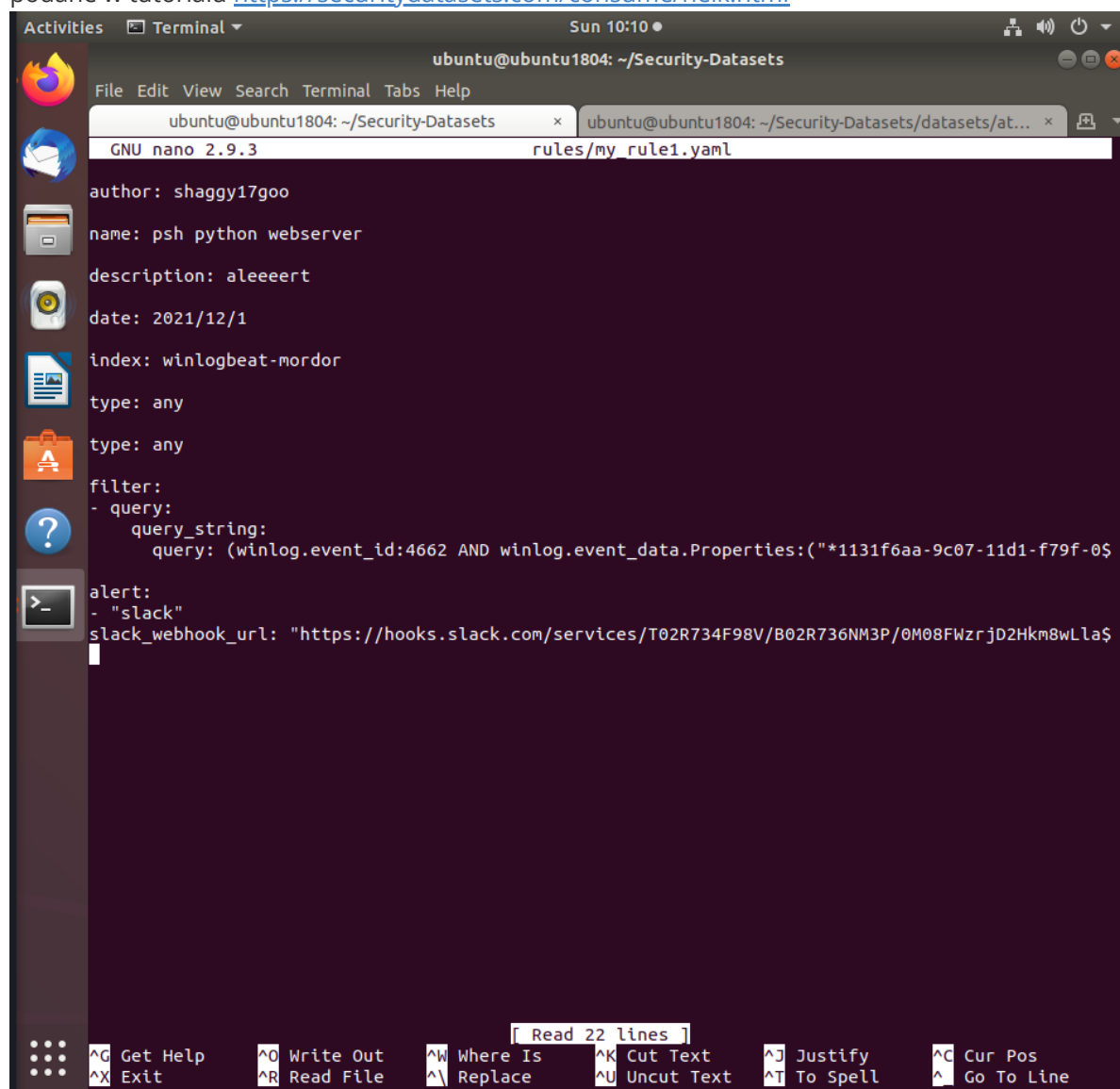


Rule

Sprawdziliśmy dostępne typy reguł wspierane przez silnik Elastalert'a. Zapoznaliśmy się z możliwościami reguł każdego typu. Postanowiliśmy stworzyć najprostszą regułę, wykrywającą sprawdzającą zawartości danych parametrów każdego z logów.



Napisana przez nas reguła wygląda w następujący sposób. Jako zapytanie KQL wykorzystaliśmy to podane w tutorialu <https://securitydatasets.com/consume/helk.html>

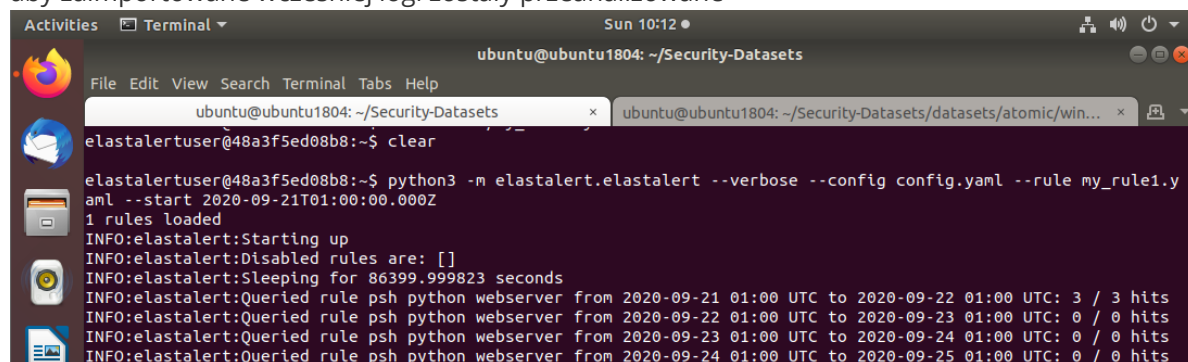


```
author: shaggy17goo
name: psh python webserver
description: aleeeert
date: 2021/12/1
index: winlogbeat-mordor
type: any
type: any
filter:
- query:
  query_string:
    query: (winlog.event_id:4662 AND winlog.event_data.Properties:("*1131f6aa-9c07-11d1-f79f-05
alert:
- "slack"
slack_webhook_url: "https://hooks.slack.com/services/T02R734F98V/B02R736NM3P/0M08FWzrjD2Hkm8wLLa$
```

W momencie spełnienia się reguły wysłany zostaje alert na Slack'a

Uruchomienie

Uruchomiliśmy silnik elasticsearch'a który sprawdza napisaną przez nas regułę. Dodatkowym problemem była konieczność wpisania daty początkowej, od której silnik ma analizować logi tak aby zaimportowane wcześniej logi zostały przeanalizowane

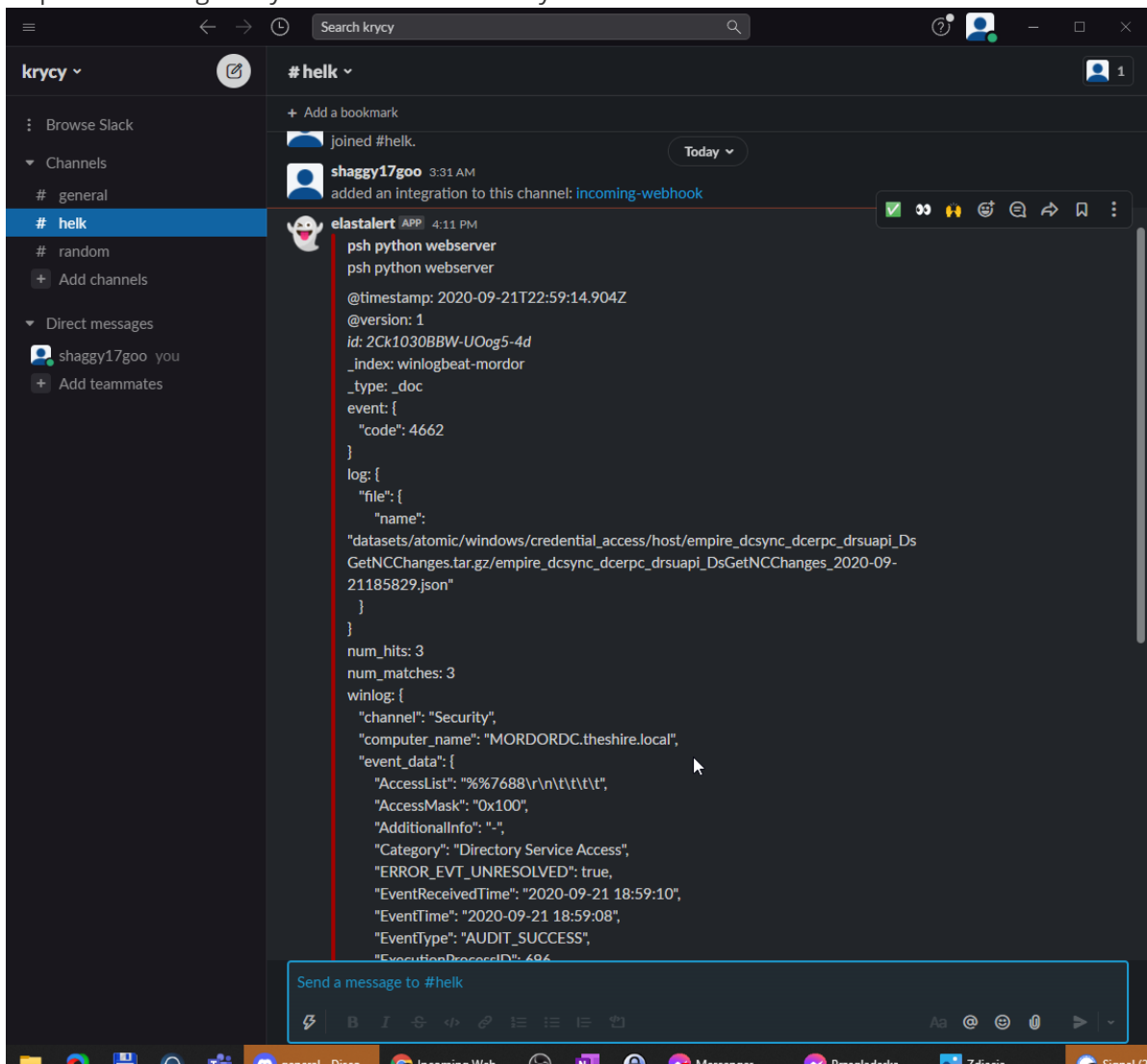


```
elastalertuser@48a3f5ed08b8:~$ clear
elastalertuser@48a3f5ed08b8:~$ python3 -m elastalert.elastalert --verbose --config config.yaml --rule my_rule1.y
aml --start 2020-09-21T01:00:00.000Z
1 rules loaded
INFO:elastalert:Starting up
INFO:elastalert:Disabled rules are: []
INFO:elastalert:Sleeping for 86399.999823 seconds
INFO:elastalert:Queried rule psh python webserver from 2020-09-21 01:00 UTC to 2020-09-22 01:00 UTC: 3 / 3 hits
INFO:elastalert:Queried rule psh python webserver from 2020-09-22 01:00 UTC to 2020-09-23 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2020-09-23 01:00 UTC to 2020-09-24 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2020-09-24 01:00 UTC to 2020-09-25 01:00 UTC: 0 / 0 hits
```

```
INFO:elastalert:Queried rule psh python webserver from 2021-12-15 01:00 UTC to 2021-12-16 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2021-12-16 01:00 UTC to 2021-12-17 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2021-12-17 01:00 UTC to 2021-12-18 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2021-12-18 01:00 UTC to 2021-12-19 01:00 UTC: 0 / 0 hits
INFO:elastalert:Queried rule psh python webserver from 2021-12-19 01:00 UTC to 2021-12-19 15:11 UTC: 0 / 0 hits
/usr/local/lib/python3.6/dist-packages/urllib3/connectionpool.py:988: InsecureRequestWarning: Unverified HTTPS request is being made to host 'hooks.slack.com'. Adding certificate verification is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#ssl-warnings
InsecureRequestWarning,
INFO:elastalert:Alert 'psh python webserver' sent to Slack
INFO:elastalert:Ignoring match for silenced rule psh python webserver
INFO:elastalert:Ignoring match for silenced rule psh python webserver
INFO:elastalert:Ran psh python webserver from 2020-09-21 01:00 UTC to 2021-12-19 15:11 UTC: 0 query hits (0 already seen), 3 matches, 1 alerts sent
```

Slack

Dopasowanie reguł i wysłanie alertów możemy zaobserwować na kanale SLACK



Model detekcji anomalii - detekcja progowa

Wybranie opcji do realizacji

Wybraliśmy opcję z przygotowaniem modelu detekcji anomalii dla ruchu sieciowego.

Propozycja przykładu anomalii

Zaproponowane metryki

W ramach tego laboratorium proponujemy aż trzy metryki anomalii:

- **Anomalia długości połączeń między hostami** - korzystając z pliku ground truth (w przypadku wybranych przez nas danych zrzutu ruchu z poniedziałku), obliczmy ile

proporcjonalnie powinno być połączeń zawierających się w danym przedziale długości okna czasowego i sprawdzamy czy w danych testowanych nie długości połączeń nie odbiegają za bardzo (0.5gt, 1.5gt) od danych z ground truth. W ten sposób mieliśmy nadzieję otrzymać alerty gdy na przykład system będzie intensywnie skanowany. Jednakże w danych, które analizowaliśmy nie udało się to - wszystkie mają podobne rozkłady czasowe - dlatego właśnie powstały dwie kolejne metryki

- **Anomalia w postaci nieznanych portów** - korzystając z pliku ground truth znajdujemy wszystkie używane numery portów i sprawdzamy czy w pliku testów nie znajdują się zapisy o połączeniach z i na inne porty (niebędące w pliku z gt). Ta analiza dla odmiany była zbyt szeroka i dawała zbyt dużo alertów.
- **Anomalie w średnich z wszystkich możliwych metryk** - bardzo ogólna analiza sprawdzająca średnią z wszystkich możliwych metryk w danych testowych. Alert jest podnoszony gdy średnia z danej metryki z danych testowanych znajduje się poza zakresem $0.1 * (\text{średnia z tej samej metryki w danych gt})$, $1.9 * (\text{średnia z tej samej metryki w danych gt})$. Ta metryka często pokazuje alerty przy wielu przykładowych atakach.

Źródło danych i faktyczne dane

Źródłem danych są przetworzone dane o ruchu sieciowym z CIC-IDS2017. Badana sieć składa się z modemu, firewall'a, kilku switchy, routerów i wielu maszyn z systemem Windows/Ubuntu lub Mac OS X. Dane zawierają zrzuty dane w formie csv z 5 dni - poniedziałek jest dniem przykładowym (w którym nie ma ataków), a przez kolejne dni następują ataki takie jak Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet i DDoS.

```
Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv
Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv
Friday-WorkingHours-Morning.pcap_ISCX.csv
Monday-WorkingHours.pcap_ISCX.csv
Thursday-WorkingHours-Afternoon-Infiltration.pcap_ISCX.csv
Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv
Tuesday-WorkingHours.pcap_ISCX.csv
Wednesday-workingHours.pcap_ISCX.csv
archive.zip
```

Notatnik

Poniżej przedstawiamy wymagania i sekcje kodu(w notatniku) je realizujące:

Wymaganie	Realizacja
ANO.PROG.3.1	<pre>## LOAD DATA ## https://www.kaggle.com/c/cicdataset/cicids2017 monday_df = pd.read_csv('data/Monday-WorkingHours.pcap_ISCX.csv', sep=',', encoding='UTF8') test_df = pd.read_csv('data/Tuesday-WorkingHours.pcap_ISCX.csv', sep=',', encoding='UTF8')</pre>
ANO.PROG.3.2	<pre>def flow_duration_analytics(data_frame): analytics_dict = {} flow_duration = data_frame[' Flow Duration'] analytics_dict['flow_duration_mean'] = np.mean(flow_duration) analytics_dict['dur_0_200'] = 0 analytics_dict['dur_200_1000'] = 0 analytics_dict['dur_1000_100000'] = 0 analytics_dict['dur_100000_1000000'] = 0 analytics_dict['dur_1000000_100000000'] = 0 analytics_dict['dur_100000000_more'] = 0 for dur in flow_duration: if dur <= 200: analytics_dict['dur_0_200'] += 1 elif dur <=1000: analytics_dict['dur_200_1000'] += 1 elif dur <=100000: analytics_dict['dur_1000_100000'] += 1 elif dur <=1000000: analytics_dict['dur_100000_1000000'] += 1 elif dur <=100000000: analytics_dict['dur_1000000_100000000'] += 1 else: analytics_dict['dur_100000000_more'] +=1 return analytics_dict def calculate_overall_analytics(data_frame): res_dict = {} for key in data_frame: try: res_dict[key] = np.mean(data_frame[key]) except: pass return res_dict def normalize_test_data(gt_analysis, test_analysis): normalized_test_analysis = {} for key in test_analysis: if key != 'sum' and key != 'flow_duration_mean': normalized_test_analysis[key] = test_analysis[key] * (gt_analysis['sum'] / test_analysis['sum']) else: normalized_test_analysis[key] = test_analysis[key] return normalized_test_analysis</pre>

Wymaganie	Realizacja
<p>ANO.PROG.3.3, ANO.PROG.3.4, ANO.PROG.3.5</p>	<pre>def find_anomaly_in_flow_duration(gt_data, test_data): gt_analysis = flow_duration_analytics(gt_data) test_analysis = flow_duration_analytics(test_data) test_analysis = normalize_test_data(gt_analysis, test_analysis) for key in gt_analysis: if not (gt_analysis[key]*0.5 < test_analysis[key] < gt_analysis[key]*1.5): print(f"ALERT {key}, gt={gt_analysis[key]}, test={test_analysis[key]}") def compare_overall_analytics(test_frame, gt_analytics): important_keys = [] for key in test_frame: try: if 0.1*gt_analytics[key] > np.mean(test_frame[key]): print(f"ALERT a lot less {key}") important_keys.append(key) elif np.mean(test_frame[key]) > 1.9*gt_analytics[key]: print(f"ALERT a lot more {key}") important_keys.append(key) except: pass return important_keys for port in unique_ports_test: if port not in unique_ports_gt: print(f"ALERT port={port}")</pre>
<p>##### Integracja analizy z notatnika z rozwiązaniem z Lab 1A</p>	
<p>W celu zintegrowania notatnika powstał nowy plik threshold_detection.py (i został dodany jako funkcja click), który był adaptacją kodu z notatnika.</p>	

Test i prezentacja działania zintegrowanego projektu:

Poniżej znajdują się wyniki analizy długości połączeń oraz średnich z wszystkich możliwych metryk:

```
(base) [kali@kali] ~/Desktop/KRYCY/scen1
~$ python blue_toolkit.py anomaly-detection -f ../data/Tuesday-WorkingHours.pcap_ISCX.csv -t flow_duration
[LOG] 2021-12-19 16:41:26.392511 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, flow_duration) - loaded test data
[LOG] 2021-12-19 16:41:26.616248 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, flow_duration) - Analysis is done

(base) [kali@kali] ~/Desktop/KRYCY/scen1
~$ python blue_toolkit.py anomaly-detection -f ../data/Tuesday-WorkingHours.pcap_ISCX.csv -t high-difference
[LOG] 2021-12-19 16:41:37.145683 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, high-difference) - loaded test data
[LOG] 2021-12-19 16:41:37.148212 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, high-difference) - Calculating all possible analytics - this may take a while
[ALERT] 2021-12-19 16:42:56.287959 compare_overall_analytics(test_frame, gt_analytics) - Fwd Header Length values are a lot smaller in tested data
[ALERT] 2021-12-19 16:42:56.291349 compare_overall_analytics(test_frame, gt_analytics) - Bwd Header Length values are a lot smaller in tested data
[ALERT] 2021-12-19 16:42:56.384261 compare_overall_analytics(test_frame, gt_analytics) - RST Flag Count values are a lot bigger in tested data
[ALERT] 2021-12-19 16:42:56.314582 compare_overall_analytics(test_frame, gt_analytics) - ECE Flag Count values are a lot bigger in tested data
[ALERT] 2021-12-19 16:42:56.321482 compare_overall_analytics(test_frame, gt_analytics) - Fwd Header Length.1 values are a lot smaller in tested data
[ALERT] 2021-12-19 16:42:56.348749 compare_overall_analytics(test_frame, gt_analytics) - min_seg_size_forward values are a lot smaller in tested data
[ERROR] 2021-12-19 16:42:56.352764 compare_overall_analytics(test_frame, gt_analytics) - unable to fetch results for key: Label
[LOG] 2021-12-19 16:42:56.355472 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, high-difference) - Analysis is done
```

Poniżej znajduje się kawałek analizy występowania połączeń z nieznanych portów. Wyników jest jeszcze więcej niż pokazane jest na rzucie ekranu, ale nie jest to bardzo zaskakujące biorąc pod

uwagę, że pliki mają po około 500000 logów:

```
(base) (kali@kali) - [~/Desktop/KRYCY/scen1]
~$ python blue_toolkit.py anomaly-detection -f ../data/Tuesday-WorkingHours.pcap_ISCX.csv -t unique_ports
[LOG] 2021-12-19 16:43:45.888684 anomaly-detection(../data/Tuesday-WorkingHours.pcap_ISCX.csv, unique_ports) - loaded test data
[ALERT] 2021-12-19 16:43:45.898175 find_anomaly_in_unique_ports(gt_data, test_data) - port=1054 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.901186 find_anomaly_in_unique_ports(gt_data, test_data) - port=1040 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.903892 find_anomaly_in_unique_ports(gt_data, test_data) - port=26654 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.906083 find_anomaly_in_unique_ports(gt_data, test_data) - port=37865 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.908500 find_anomaly_in_unique_ports(gt_data, test_data) - port=43970 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.912454 find_anomaly_in_unique_ports(gt_data, test_data) - port=48231 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.916035 find_anomaly_in_unique_ports(gt_data, test_data) - port=1073 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.918669 find_anomaly_in_unique_ports(gt_data, test_data) - port=1317 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.920865 find_anomaly_in_unique_ports(gt_data, test_data) - port=1315 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.922975 find_anomaly_in_unique_ports(gt_data, test_data) - port=1339 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.925510 find_anomaly_in_unique_ports(gt_data, test_data) - port=1397 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.928392 find_anomaly_in_unique_ports(gt_data, test_data) - port=1348 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.933365 find_anomaly_in_unique_ports(gt_data, test_data) - port=1507 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.937065 find_anomaly_in_unique_ports(gt_data, test_data) - port=1379 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.939287 find_anomaly_in_unique_ports(gt_data, test_data) - port=1362 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.942101 find_anomaly_in_unique_ports(gt_data, test_data) - port=1399 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.945333 find_anomaly_in_unique_ports(gt_data, test_data) - port=1572 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.949219 find_anomaly_in_unique_ports(gt_data, test_data) - port=1605 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.951900 find_anomaly_in_unique_ports(gt_data, test_data) - port=1612 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.954382 find_anomaly_in_unique_ports(gt_data, test_data) - port=1570 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.956497 find_anomaly_in_unique_ports(gt_data, test_data) - port=1581 present in tested data but not in ground truth data
[ALERT] 2021-12-19 16:43:45.958612 find_anomaly_in_unique_ports(gt_data, test_data) - port=1603 present in tested data but not in ground truth data
```

Wykorzystanie znanych rozwiązań dla reguł YARA

Wymaganie	Realizacja
REG.DET.1	YARA
REG.DET.2.1	<pre>(DNS) [osboxes@osboxes scen1]\$ python3 blue_toolkit.py list-yara-rules -r /home/osboxes -d Found YARA rules: 1. crypto_signatures.yar 2. rogue_ip.yar 3. dummy_rule.yar (DNS) [osboxes@osboxes scen1]\$ python3 blue_toolkit.py scan-files-with-yara-rules --help Usage: blue_toolkit.py scan-files-with-yara-rules [OPTIONS] Options: -r, --rules TEXT Path to directory with YARA rules / Path to single rule [required] -D, --rules-deep Search for rules in subdirectories -s, --rule-selection TEXT Comma separated numbers of rules to use (e.g. -r 1,3,4), by default all available rules from are used -p, --path TEXT Path to file or directory to scan [required] -d, --deep Scan files in subdirectories -t, --type [txt json xml evtx] File type to load. By default all types are loaded --help Show this message and exit.</pre>
REG.DET.2.2	<pre>(DNS) [osboxes@osboxes scen1]\$ python3 server.py INFO: Will watch for changes in these directories: ['/home/osboxes/Desktop/KRYCY/scen1'] INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit) INFO: Started reloader process [3445] using statreload INFO: Started server process [3447] INFO: Waiting for application startup. INFO: Application startup complete. ALERT name: YARA rulescontent: YARA rule RogueIP match file /home/osboxes/Desktop/test/folder3/tajneip.txt: >Found \$localhost string (b'127.0.0.1') in file /home/osboxes/Desktop/test/folder3/tajneip.txt (offset=0) []</pre>
REG.DET.3.1	N/D
REG.DET.3.2	<pre>(DNS) [osboxes@osboxes scen1]\$ python3 blue_toolkit.py scan-files-with-yara-rules -r /home/osboxes -D -s 1,2 -p /home/osboxes -d -t txt scan_files_with_yara_rules(/home/osboxes, True, 1,2, ('/home/osboxes/'), True, ('txt')) Files loaded for analysis: ext:0 json:0 xml:0 pcap:0 txt:307 Found YARA rules: 1. crypto_signatures.yar 2. rogue_ip.yar 3. dummy_rule.yar ALERT - YARA rules YARA rule BigNumbers3 match file /home/osboxes/.vscode/extensions/ms-python.python-2021.12.1559732655/requirements.txt: >Found \$c0 string (b'1a19ccace2e8b10b49430b74a3e6bf07b2f81301b0ab05c1d4433272d76f') in file /home/osboxes/.vscode/extensions/ms-python.python-2021.12.1559732655/requirements.txt (offset=169) >Found \$c0 string (b'e52ff6d3012b3162c6f9f6c51c70db7c8192ee7a3ac71e162971b9455') in file /home/osboxes/.vscode/extensions/ms-python.python-2021.12.1559732655/requirements.txt (offset=254) YARA rule BigNumbers0 match file /home/osboxes/mozilla/firefox/k7b03p3.default-release/AlternateServices.txt: >Found \$c0 string (b'14841811162018713838') in file /home/osboxes/mozilla/firefox/k7b03p3.default-release/AlternateServices.txt (offset=3775) >Found \$c0 string (b'14841811162018713838') in file /home/osboxes/mozilla/firefox/k7b03p3.default-release/AlternateServices.txt (offset=3877) >Found \$c0 string (b'14841811162018713838') in file /home/osboxes/mozilla/firefox/k7b03p3.default-release/AlternateServices.txt (offset=3919) YARA rule SigHashBigEndianConstants match file /home/osboxes/Desktop/file.txt: >Found \$c0 string (b'uespmos') in file /home/osboxes/Desktop/file.txt (offset=35) >Found \$c1 string (b'modnared') in file /home/osboxes/Desktop/file.txt (offset=54) >Found \$c2 string (b'arenegyl') in file /home/osboxes/Desktop/file.txt (offset=78) YARA rule BASE64_table match file /home/osboxes/Desktop/test/folder1/folder3/dump.txt: >Found \$c0 string (b'ABKDFGHIJLKNOPQRSTUWVYZabdcfghijklmnopqrstuvwxyZ0123456789+') in file /home/osboxes/Desktop/test/folder1/folder3/dump.txt (offset=12826) YARA rule RogueIP match file /home/osboxes/Desktop/test/folder3/tajneip.txt: >Found \$localhost string (b'127.0.0.1') in file /home/osboxes/Desktop/test/folder3/tajneip.txt (offset=0)</pre>

