

Autorzy:

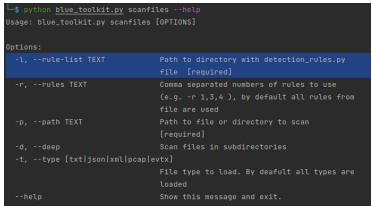
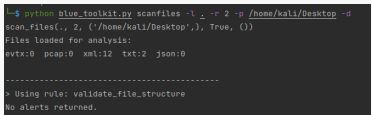
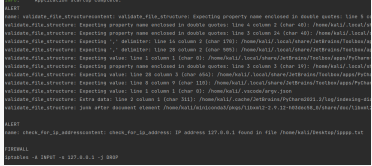
- Borkowski Mateusz
- Gryka Paweł
- Popiołek Paweł
- Wawrzyńczak Michał

LAB 1a

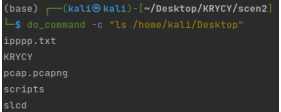
Scenariusz 1

ID wymagania	Opis wymagania	Zrzut ekranu z krótkim opisem
OFF.1	Opracować aplikację z interfejsem CLI, która pozwoli na realizację wskazanych wymagań. Moduł do tworzenia aplikacji CLI	<pre>\$ python blue_toolkit.py --help Usage: blue_toolkit.py [OPTIONS] COMMAND [ARGS]... Options: --help Show this message and exit. Commands: grepiagrep grepiare listrules printpcap scanfiles</pre>
OFF.2	Aplikacja działania w trybie CLI z wypisywaniem akcji	<pre>\$ python blue_toolkit.py grepiaregrep -f detection_rules.py -p str def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') description += (f'validate_file_structure: {a.args[0]} {json_file}\n')</pre>
OFF.2.1	Automatycznie tworzony jest log z działania o nazwie składającej się z nazwy aplikacji oraz znacznika czasu jej uruchomienia	<pre>2021-11-23 11:37:56.288947 grep_via_grep(detection_rules.py, str) 2021-11-23 11:37:56.288947 grep_via_grep(detection_rules.py, str) [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a.args[0]} {json_file}\n')</pre>
OFF.2.2	Efekty operacji są wypisywane jednocześnie na CLI oraz do otwartego pliku loga	<pre>2021-11-23 11:37:56.288947 grep_via_grep(detection_rules.py, str) 2021-11-23 11:37:56.288947 grep_via_grep(detection_rules.py, str) [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a}: {json_file}\n') [console output]: def validate_file_structure(*args): description += (f'validate_file_structure: {a.args[0]} {json_file}\n')</pre>
OFF.2.3	Aplikacja utrzymuje także bazę zdarzeń - w bazie danych SQL	<pre>68 ERROR grep_via_grep(main.py, main) - 2021-11-23 15:25:29.528388 69 ERROR grep_via_grep(main.py, main) - 2021-11-23 15:25:29.528388 70 ERROR grep_via_grep(main.py, main) - 2021-11-23 15:25:29.528388 71 LOG grep_via_grep(main.py, main) - 2021-11-24 18:50:07.558036 72 LOG grep_via_grep(main.py, main) - 2021-11-24 18:50:07.558036 73 LOG grep_via_grep(blue_toolkit.py, - 2021-11-24 18:50:07.558036 74 LOG grep_via_grep(blue_toolkit.py, - 2021-11-24 18:50:07.558036 75 LOG extract_traffic(dup.pcap, 1 - 2021-11-25 07:02:04.519086 76 LOG extract_traffic(dup.pcap, 1 - 2021-11-25 07:02:04.519086 77 LOG grep_via_grep(start_generator.py, 2021-11-25 07:02:12.802286 78 LOG grep_via_grep(blue_toolkit.py, - 2021-11-25 07:02:12.802286 79 LOG grep_via_grep(blue_toolkit.py, - 2021-11-25 07:02:12.802286 80 ERROR grep_via_grep(blue_toolkit.py, - 2021-11-25 07:02:12.802286 81 LOG extract_traffic(war/krvcs/pcap, 2021-11-25 07:06:03.238235 82 LOG extract_traffic(war/krvcs/pcap, 2021-11-25 07:06:04.238235 83 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 84 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 85 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 86 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 87 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 88 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254 89 LOG grep_via_grep(detection_rules.py, 2021-12-01 18:02:18.483254</pre>
OFF.3	Aplikacja ma możliwość wskazania pojedynczych plików, folderu lub grupy folderów do przeszukania w poszukiwaniu plików, które mają być wykorzystane do analizy	<pre>\$ python blue_toolkit.py scanfiles --help Usage: blue_toolkit.py scanfiles [OPTIONS] Options: -l, --rule-list TEXT Path to directory with detection_rules.py file [required] -r, --rules TEXT Comma separated numbers of rules to use (e.g. -r 1,3,4), by default all rules from file are used -p, --path TEXT Path to file on directory to scan [required] -d, --deep Scan files in subdirectories -t, --type [txt json xml pcap evtx] file type to load. By default all types are loaded --help Show this message and exit.</pre>
OFF.4	Obsługiwane formaty wejściowe plików do analizy: - pliki tekstowe w formatach .txt, .xml, .json - pliki ze zrzutami ruchu PCAP .pcap - pliki logów Sysmon Windows EVTX - .evtx	<pre>\$ python blue_toolkit.py scanfiles -l -r 3 -p /home/kali -d scan_files(, 3, ('/home/kali',), True, ()) Files loaded for analysis: pcap:53 txt:1182 evtx:0 xml:162 json:2891 ----- Using rule: validate_text_encoding</pre>

ID wymagania	Opis wymagania	Zrzut ekranu z krótkim opisem
OFF.5	Aplikacja ma możliwość wyświetlania zawartości pakietów z wczytanych danych z plików PCAP.	
OFF.6	Aplikacja ma możliwość przekazania filtra zgodnego z formatem BPF (wykorzystywanego przez libpcap / tshark / pyshark / Wireshark / Scapy) do funkcji otwierającej i wczytującej plik PCAP.	
OFF.7	Aplikacja ma możliwość wywołania operacji systemowej grep na wskazanych plikach tekstowych. Argumentem przekazywanym do operacji jest właściwe wyrażenie regularne.	
OFF.8	Aplikacja ma możliwość wywołania działania wyrażenia regularnego z modułu Python re . Argumentem przekazywanym do operacji jest właściwe wyrażenie regularne.	
OFF.9	Aplikacja ma możliwość załadowania reguł analitycznych do detekcji zdarzeń opisanych za pomocą tych reguł i przechowywania ich w wybranej strukturze danych.	
OFF.9.1	Reguły będą opisywane jako funkcje Pythona w pliku detection-rules.py (nazwa na sztywno)	
OFF.9.2	Każda reguła ma być zdefiniowana jako oddzielna funkcja w języku Python we wskazanym pliku w OFF.9.1 . Format pojedynczej reguły:	

ID wymagania	Opis wymagania	Zrzut ekranu z krótkim opisem
OFF.9.3	Ładowanie reguł ma odbywać się na żądanie, tj. po wywołaniu żądania ładowania - tj. plik z regułami ma nie być załadowany statycznie od początku działania aplikacji OFF.9.4 Każdorazowe wywołanie ładowania reguł ma oznaczać usunięcie i	
OFF.9.4	Każdorazowe wywołanie ładowania reguł ma oznaczać usunięcie istniejących w pamięci programu i załadowanie nowego zestawu reguł	Tak się dzieje
OFF.9.5	Informacjami zwrotnymi z reguły są: - action_alert - action_block - description	
OFF.10	Interfejs wywołania reguł analitycznych umożliwia ich użycie w kilku trybach:	
OFF.10.1	Wywołanie całego zestawu reguł na wybranym zestawie plików	
OFF.10.2	Wywołanie wybranej reguły - poprzez wskazanie jej nazwy (nazwa funkcji z OFF.9.2) - i na wybranym zestawie plików przekazanych do reguły	
OFF.11	Przygotować aplikację CLI (nie musi być oparta na Click jak w Wymaganiu OFF.1) do odbierania wiadomości po REST API i wypisywaniu ich na CLI - alert.	
OFF.12	Przygotować aplikację CLI (nie musi być oparta na Click jak w Wymaganiu OFF.1) do odbierania wiadomości po REST API i wypisywaniu ich na CLI - akcja dla firewalla.	
OFF.12.1	* (opcjonalnie) Można wykonać prostą integrację np. z iptables, ufw czy w inny proponowany sposób z firewallem w wybranym hoście.	

Scenariusz 2

ID wymagania	Opis wymagania	Zrzut ekranu z krótkim opisem
ON.MAIN.3.2	Przekazanie komendy do wykonania na zdalnym hoście i odebranie odpowiedzi	 <pre>(base) root@kali: ~/Desktop/KRYCY/scen2 ~\$ do_command -c "ls /home/kali/Desktop" ipppp.txt KRYCY pcap.pcapng scripts slcd</pre>

Testy

ID testu	Opis testu	Zrzut ekranu z krótkim opisem
1	Prawidłowe wczytywanie każdego z 5 typów danych do analizy.	<pre> L\$ python blue_toolkit.py scanfiles -l _ -r 2 -p /home/kali/ -d scan_files(., 2, ('/home/kali/',), True, ()) Files loaded for analysis: json:2892 evtx:1 txt:1188 pcap:53 xml:162 </pre>
2	Działanie filtra przy wczytywaniu pliku PCAP.	<pre> L\$ python blue_toolkit.py printpcap -p /home/kali/Desktop/pcap.pcapng Display filter []: arp Packet (Length: 60) Layer ETH: Destination: ff:ff:ff:ff:ff:ff Address: ff:ff:ff:ff:ff:ff 1. = LG bit: Locally administered address (this is NOT the factory default) 1. = IG bit: Group address (multicast/broadcast) Source: 00:50:56:c0:00:08 0. = LG bit: Globally unique address (factory default) 0. = IG bit: Individual address (unicast) Type: ARP (0x0806) Padding: 00000000000000000000000000000000 Address: 00:50:56:c0:00:08 Layer ARP: Hardware type: Ethernet (1) Protocol type: IPv4 (0x0800) Hardware size: 6 Protocol size: 4 Opcode: request (1) Sender MAC address: 00:50:56:c0:00:08 Sender IP address: 192.168.112.1 Target MAC address: 00:00:00:00:00:00 Target IP address: 192.168.112.2 Packet (Length: 60) </pre>
3	Działanie grep .	<pre> L\$ python blue_toolkit.py grepviagrep -f detection_rules.py -p '\:\$' def validate_text_encoding(**kwargs): for txt_file in kwargs['txt']: for encoding in ENCODING_LIST: try: except: if failed == len(ENCODING_LIST): if not detection: def validate_file_structure(**kwargs): for json_file in kwargs['json']: try: except Exception as e: for xml_file in kwargs['xml']: try: except Exception as e: if not detection: def check_for_ip_address(**kwargs): </pre>
4	. Działanie wyrażenia regularnego z modułem re .	<pre> L\$ python blue_toolkit.py grepviare -f detection_rules.py -p '^[di]' import json import os import re import xml.sax import Evtx.Evtx as evtx import nest_asyncio import pyshark as ps def validate_text_encoding(**kwargs): def validate_file_structure(**kwargs): def check_for_ip_address(**kwargs): </pre>
5	Wczytanie reguł detekcyjnych z pliku detection-rules.py .	<pre> L\$ python blue_toolkit.py listrules -r _ Detection rules in provided file: 1. check_for_ip_address 2. validate_file_structure 3. validate_text_encoding </pre>

ID testu	Opis testu	Zrzut ekranu z krótkim opisem
6	Wykonanie reguły detekcyjnej - wszystkich (co najmniej 2), pojedynczej po nazwie.	<pre> \$ python blue_toolkit.py scanfiles -l _ -p /home/kali/Desktop -d scan_files(., None, ('/home/kali/Desktop',), True, ()) Files loaded for analysis: txt:2 evtx:0 pcap:0 xml:12 json:0 ----- > Using rule: check_for_ip_address Scanning for IPs: ['127.0.0.1', '10.0.0.1'] ALERT (1) - check_for_ip_address check_for_ip_address: IP address 127.0.0.1 found in file /home/kali/Desktop/ipppp.txt ----- > Using rule: validate_file_structure No alerts returned. ----- > Using rule: validate_text_encoding No alerts returned. \$ python blue_toolkit.py scanfiles -l _ -r 2 -p /home/kali/Desktop -d scan_files(., 2, ('/home/kali/Desktop',), True, ()) Files loaded for analysis: evtx:0 pcap:0 xml:12 txt:2 json:0 ----- > Using rule: validate_file_structure No alerts returned. </pre>
7	Zaprezentowanie akcji alert-local, alert-remote oraz block.	<pre> INFO: Application startup complete. ALERT name: validate_file_structurecontent: validate_file_structure: Expecting property name enclosed in double quotes: line 5 c validate_file_structure: Expecting property name enclosed in double quotes: line 4 column 2 (char 40): /home/kali/.local/s validate_file_structure: Expecting property name enclosed in double quotes: line 3 column 24 (char 40): /home/kali/.local/ validate_file_structure: Expecting '.' delimiter: line 14 column 2 (char 170): /home/kali/.local/share/JetBrains/Toolbox/a validate_file_structure: Expecting '.' delimiter: line 28 column 2 (char 595): /home/kali/.local/share/JetBrains/Toolbox/a validate_file_structure: Expecting value: line 1 column 1 (char 0): /home/kali/.local/share/JetBrains/Toolbox/apps/PyCharm validate_file_structure: Expecting property name enclosed in double quotes: line 3 column 3 (char 19): /home/kali/.local/s validate_file_structure: Expecting value: line 28 column 3 (char 654): /home/kali/.local/share/JetBrains/Toolbox/apps/PyCh validate_file_structure: Expecting value: line 8 column 9 (char 110): /home/kali/.local/share/JetBrains/Toolbox/apps/PyCharm validate_file_structure: Extra data: line 2 column 1 (char 311): /home/kali/.cache/JetBrains/PyCharm2021.2/Log/indexing-di validate_file_structure: junk after document element /home/kali/.miniconda3/pkgs/libxml2-2.9.12-h03d0c58_0/share/doc/libxml ALERT name: check_for_ip_addresscontent: check_for_ip_address: IP address 127.0.0.1 found in file /home/kali/Desktop/ipppp.txt FIREWALL iptables -A INPUT -s 127.0.0.1 -j DROP </pre>
8	Zdalne uruchomienie zbierania pliku PCAP i pobranie do aplikacji głównej.	<pre> (base) (kali@kali) - [~/Desktop/KRYCY/scen2] \$ do_sniff -i eth0 -t 10 -n pcap.pcap (base) (kali@kali) - [~/Desktop/KRYCY/scen2] \$ ls agent.py main_app.py main.egg-info pcap.pcap __pycache__ setup.py (base) (kali@kali) - [~/Desktop/KRYCY/scen2] \$ ll total 36 -rw-r--r-- 1 kali kali 1925 Nov 25 06:59 agent.py -rw-r--r-- 1 kali kali 1705 Nov 25 06:59 main_app.py drwxr-xr-x 2 kali kali 4096 Dec 1 12:34 main.egg-info -rw-r--r-- 1 kali kali 16384 Dec 1 12:38 pcap.pcap drwxr-xr-x 2 root root 4096 Dec 1 12:34 __pycache__ -rw-r--r-- 1 kali kali 543 Nov 25 06:59 setup.py </pre>

ID testu	Opis testu	Zrzut ekranu z krótkim opisem
9	Zdalne pobranie logów systemowych.	<pre> (base) └─(kali㉿kali)-[~/Desktop/KRYCY/scen2] └─\$ getLogList ['log.1', 'looog.log.1.log'] (base) └─(kali㉿kali)-[~/Desktop/KRYCY/scen2] └─\$ getLog -f log.1 -f looog.log.1.log (base) └─(kali㉿kali)-[~/Desktop/KRYCY/scen2] └─\$ ll total 56 -rw-r--r-- 1 kali kali 1925 Nov 25 06:59 agent.py -rw-r--r-- 1 kali kali 13 Dec 1 12:45 log.1 -rw-r--r-- 1 kali kali 0 Dec 1 12:45 looog.log.1.log -rw-r--r-- 1 kali kali 1705 Nov 25 06:59 main_app.py drwxr-xr-x 2 kali kali 4096 Dec 1 12:34 main.egg-info -rw-r--r-- 1 kali kali 16384 Dec 1 12:38 pcap.pcap drwxr-xr-x 2 root root 4096 Dec 1 12:34 __pycache__ -rw-r--r-- 1 kali kali 16384 Dec 1 12:40 pyshark.3.pcap -rw-r--r-- 1 kali kali 543 Nov 25 06:59 setup.py </pre>
10	Zdalne wykonanie komendy systemowej i pobranie jej wyniku do wyświetlenia	<pre> (base) └─(kali㉿kali)-[~/Desktop/KRYCY/scen2] └─\$ do_command -c "ls /home/kali/Desktop" ipppp.txt KRYCY pcap.pcapng scripts slcd </pre>