

Politechnika Warszawska

W Y D Z I A Ł E L E K T R Y C Z N Y



Praca dyplomowa inżynierska

na kierunku Informatyka Stosowana
w specjalności Inżynieria Oprogramowania

Oprogramowanie wykrywające ataki typu ransomware na podstawie analizy statystyk
generowanych przez system plików

Maciej Michalski

numer albumu 311351

promotor

dr inż. Radosław Roszczyk

WARSZAWA 2023

Oprogramowanie wykrywające ataki typu ransomware na podstawie analizy statystyk generowanych przez system plików

Streszczenie

Poniższa praca "Oprogramowanie wykrywające ataki typu ransomware na podstawie analizy statystyk generowanych przez system plików" zajmuje się tematyką ochrony danych zamieszczonych na infrastrukturze cyfrowej, która znajduje odzwierciedlenie w szeroko pojętej dziedzinie administracji systemów oraz cyberbezpieczeństwa. W związku ze stopniowo narastającym zagrożeniem naruszenia infrastruktury sieciowej szerokiej gamy instytucji należy w sposób sumienny i pragmatyczny ocenić, praktyczność nowych i istniejących środków zaradczych dla tego typu ataków. Głównym celem pracy było utworzenie „proof of concept” aplikacji wykrywającej operacje na systemie plików i analizującej je pod kątem ataku ransomware. Praca skupia się na analizie technicznej i merytorycznej możliwych do wykorzystania rozwiązaniach technicznych wraz z ich zaletami i wadami. Rozwiązanie zostało przetestowane na specjalnym środowisku mającym na celu odwzorowanie rodzaju infrastruktury stosowanej w rzeczywistych warunkach produkcyjnych. Kombinacja zaimplementowanych rozwiązań znalazła swoje zastosowanie w każdym ze scenariuszy testowych, w tym rzeczywistych ataków oprogramowaniem ransomware.

Słowa kluczowe: ransomware, administracja, cyberbezpieczeństwo

Software to detect ransomware attacks based on analysis of statistics generated by the file system

Abstract

Here will be an abstract of the paper Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Keywords: ransomware, administration, cybersecurity

Spis treści

1	Wprowadzenie	9
1.1	Cel pracy	9
1.2	Opis problemu i znaczenie zagrożeń typu ransomware	10
1.3	Krótką charakterystyka ataków ransomware	13
2	Przegląd literatury	17
2.1	Historia i ewolucja ataków typu ransomware	17
2.1.1	Wczesna historia	17
2.1.2	Historia współczesna	18
2.2	Istniejące techniki wykrywania i obrony przed ransomware	22
2.2.1	Wykrywanie poprzez sygnaturę plików	22
2.2.2	Wykrywanie poprzez analizę zachowania systemu	23
2.2.3	Wykrywanie poprzez analizę ruchu sieciowego	24
2.3	Podstawy działania systemów plików	24
2.3.1	Skrócony opis działania systemu plików	24
2.3.2	Monitorowanie zmian na systemie plików	26
2.3.3	Krótką charakterystyka plików wykonywalnych	27
2.4	Metody analizy statystyk systemu plików	29
2.4.1	Analiza entropii pliku	29
2.4.2	Automatyczna analiza behawioralna poprzez audyt systemu	32
2.4.3	Analiza podobieństwa pliku wykonywalnego	34
3	Analiza problemu	37
3.1	Charakterystyka typowych zmian w systemie plików podczas ataku ransomware	37
3.2	Wybór odpowiednich statystyk i metryk do analizy	38
3.3	Potencjalne wyzwania i ograniczenia metody	39
4	Projekt oprogramowania i użyte rozwiązania	41
4.1	Specyfikacja wymagań funkcjonalnych i нефункциональных	41
4.1.1	Wymagania funkcjonalne	41
4.1.2	Wymagania jakościowe	42

4.2	Sposób zbierania i przetwarzania statystyk systemu plików	44
4.3	Wybór technologii i narzędzi programistycznych	46
4.3.1	Zbieranie informacji z audytu	46
4.3.2	Logika biznesowa	46
4.3.3	Baza danych	47
4.4	Architektura aplikacji	47
4.5	Zabezpieczenia i uwierzytelnianie w systemie	49
4.6	Implementacja algorytmów wykrywających podejrzaną działalność	49
5	Testowanie i walidacja	51
5.1	Metodologia testowania	51
5.1.1	Warunki testowe i zastosowane środki bezpieczeństwa	51
5.1.2	Ustalenie metryki skuteczności rozwiązania	52
5.2	Scenariusze testowe symulujące ataki ransomware	52
5.2.1	Improwizowany atak z kompresowaniem	52
5.2.2	Atak Ransom EXX	54
5.2.3	Atak Hive	56
5.3	Ewaluacja skuteczności wykrywania	57
6	Podsumowanie	59
6.1	Główne osiągnięcia pracy	59
6.2	Ograniczenia proponowanej metody	59
6.3	Propozycje dalszego rozwoju i doskonalenia systemu	60
	Bibliografia	61
	Spis rysunków	65
	Spis tabel	67
	Spis załączników	69

Rozdział 1

Wprowadzenie

Głównym celem każdej aplikacji cyfrowej, systemu informatycznego czy innego rodzaju usług dostępnych przez sieć jest przetwarzanie informacji cyfrowej. Dzięki dygitalizacji usług oraz utworzeniu kompletnie nowych jej rodzajów zależnych od technologii cyfrowych ludzkość generuje masywne ilości danych każdego dnia. Jednym z najpopularniejszych środków komunikacji, zwłaszcza dla biznesu, jest poczta elektroniczna. Grupa **Radicati Inc.** spekuluje, że do końca 2023 liczba wysłanych listów elektronicznych powinna przekroczyć 347 miliardów [1]. **Domo, Inc.**, które jest jednym z wielu dostawców usług chmurowych, w swoim raporcie zatytułowanym „Data Never Sleeps 10.0” donosi o tym, że wielkość danych, które zostaną utworzone czy skopiowane może wejść w okolice 181 zettabajtów¹ wielkości do roku 2025 [2]. Znakomita część tych danych musi być przechowywana na stałe, gdyż wymaga tego poprawne działanie systemu lub może wynikać z nakazów prawnych. Z tych powodów jednym z priorytetów przy administracji systemu jest zabezpieczenie przed utratą danych przez instytucje i działalności gospodarcze. Do powodów utraty danych mogą należeć:

- awaria nośników i innych elementów,
- niespodziewane braki w dostawach prądu,
- błąd ludzki,
- wirusy komputerowe.

1.1 Cel pracy

Celem tej pracy było odnalezienie sposobu na zniwelowanie strat danych w wyniku ataku wirusa typu ransomware możliwego do wykorzystania przez administratorów w warunkach rzeczywistego ataku. Zaproponowanym przeze mnie rozwiązaniem jest „proof of concept” w postaci oprogramowania analizującego działania na plikach dla systemów operacyjnych z rodziny Linux. Program korzysta z informacji o stanie systemu plików i na bieżąco analizuje wykonywane na nim operacje. Statystyki z obserwowanego obszaru zawierają w sobie m.in. ilości operacji, ścieżkę do użytej komendy, nazwę użytkownika, który dokonuje operacji etc. Dzięki temu administrator może nie tylko dowiedzieć

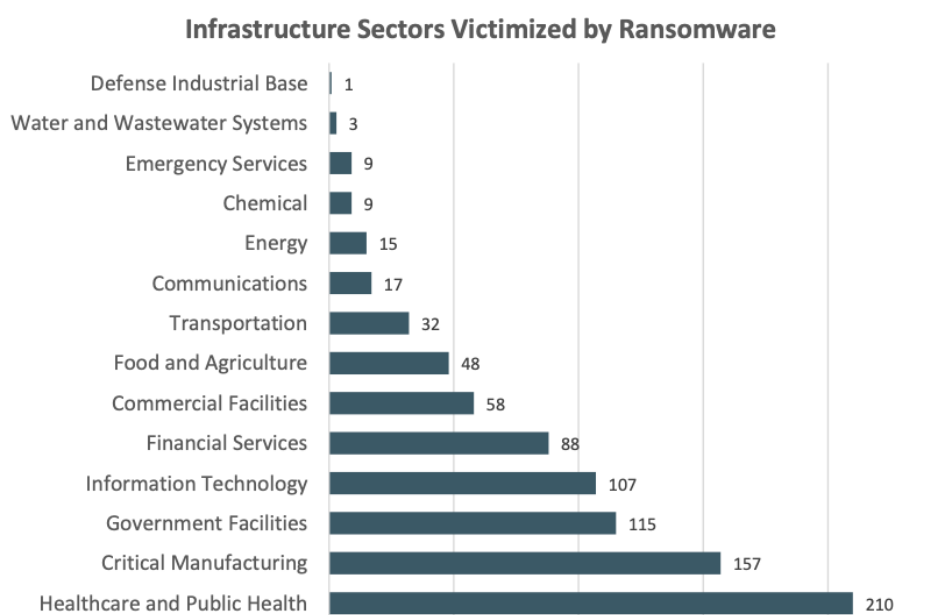
¹Zetta bajt (skrót **ZB**) w systemie SI to tryliard 10^{21} bajtów i 2^{70} , czyli 1024^7 bajtów.

się o potencjalnym zagrożeniu ataku ransomware, ale też obserwować dowolny, podejrzany ruch na systemie plików. Następnie dokonywana jest analiza zawartości plików i generowany jest raport o zakresie ryzyka. Docelową grupą użytkowników są administratorzy, a więc główne założenia, jakie postawiłem sobie w trakcie tworzenia rozwiązania, miały na celu wytworzenie oprogramowania łatwego dla nich w obsłudze. Tymi założeniami są:

- łatwa instalacja, która nie wymaga aktualizacji sterowników sprzętowych,
- wsparcie dla najpopularniejszych dystrybucji serwerowych²,
- minimalne zużycie zasobów,
- integracja z bieżącymi popularnymi rozwiązaniami w administracji systemów.

1.2 Opis problemu i znaczenie zagrożeń typu ransomware

Od 2017 roku obserwuje się trend wzrostowy ataków ransomware[3], a w ciągu pierwszej połowy 2022 roku, dokonano 236,7 miliona ataków na całym świecie [4]. Wg. raportu Verizona³ ataki ransomware stanowią 10% wszystkich naruszeń danych w 2021. Wedle zebranych statystyk wykrycie ich zajęło w aż 49 dni dłużej niż średni czas wykrycia wszystkich naruszeń z tego samego roku. Raport wyjaśnia też, że zagrożona nie jest wyłącznie branża IT, ale też inne sektory, w szczególności sektor ochrony zdrowia.



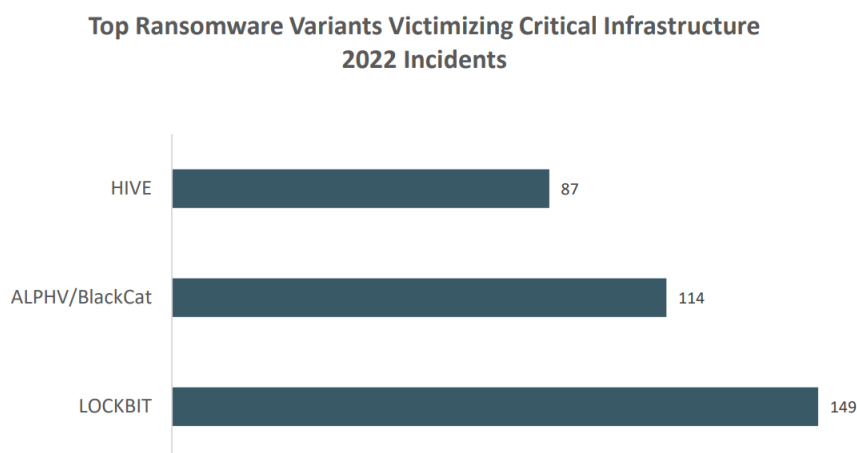
Rysunek 1. Sektory infrastruktury krytycznej, do których odnosiły się skargi IC3⁴.

²W3 Techs utrzymuje raport o sieciowych serwerach Linuksowych. Jest on codziennie aktualizowany i można go odnaleźć pod adresem: <https://w3techs.com/technologies/details/os-linux>

³Raport jest odpłatnie dostępny pod linkiem: <https://www.verizon.com/business/resources/reports/dbir/2021/results-and-analysis>

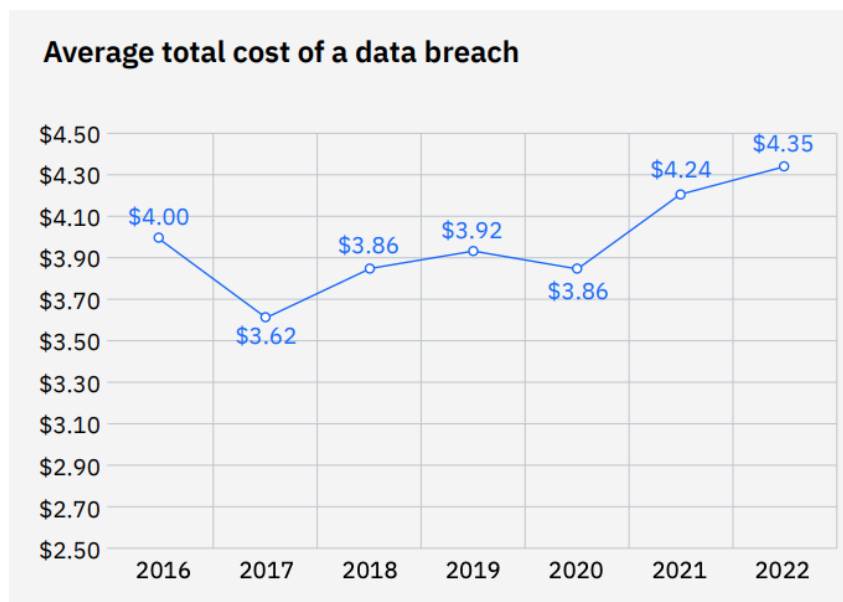
⁴FBI 2022 Internet Crime Report, s. 14

Raport IC3 z roku 2022 donosi o 870 zarejestrowanych skargach dotyczących ataków, których celem były organizacje infrastruktury krytycznej. Pośród 16 sektorów, 14 z nich padło ofiarą próby ataku.



Rysunek 2. Najpopularniejsze warianty wirusów ransomware, zarejestrowane w trakcie incydentów mających na celu atak infrastruktury krytycznej. Należy zauważyć, że wirus „LockBit” sprawiał najwięcej problemów. Jego wersja na system Linux nosi nazwę „LockBit Linux-ESXi Locker”⁵.

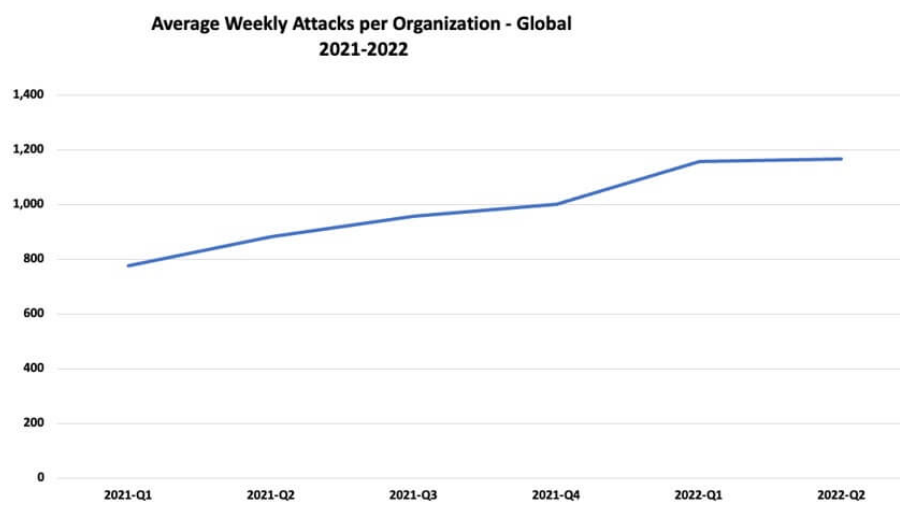
Raport grupy „Herjavec” donosi, że aż 70% organizacji medycznych borykało się z poważnymi komplikacjami przez ataki ransomware [5]. W 2022 roku 1 na 42 instytucje ochrony zdrowia były ofiarami tychże ataków, 74% z nich to szpitale [6].



Rysunek 3. Średni koszt naruszenia danych 2016-2022⁶.

⁵FBI 2022 Internet Crime Report, s. 15

Z danych zebranych z ostatnich 5 lat jednoznacznie wynika, że nieumiejętne przeciwdziałanie może zaszkodzić nie tylko finansom zaatakowanej działalności lub osoby indywidualnej, ale również stwarza zagrożenie dla zdrowia i życia. Dodatkowo, mając na uwadze średni koszt naruszenia danych w 2023, którego globalna średnia wynosi 4,45 milionów USD [7], coraz więcej administratorów jest zmuszonych dywersyfikować sposoby zabezpieczania systemów.



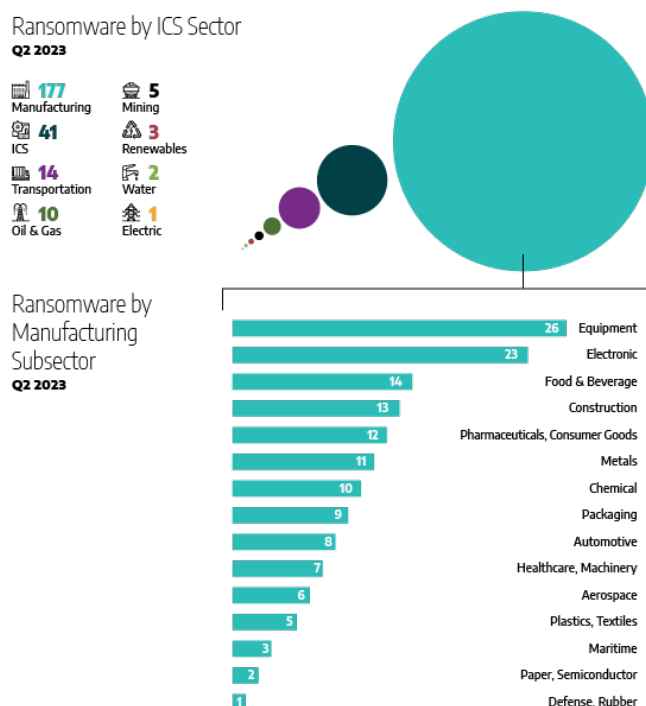
Rysunek 4. Globalnie zgłoszone incydenty ataków ransomware per kwartał w roku 2022 zarejestrowanych przez Check Point Research. Organizacja spekuluje, że wzrost ataków mógł być spowodowany lukami bezpieczeństwa „log4j” oraz cyberataków związanych z wojną w Ukrainie⁷.

Na rynku istnieje wiele popularnych rozwiązań działających prewencyjnie m.in. w tym rozbudowane aplikacje służące do tworzenia i przywracania kopii zapasowych. Należy jednak wziąć pod uwagę, że przywracanie danych nie jest prostym procesem. W zależności od rodzaju użytego nośnika przywracanie może doprowadzić nawet do przypadkowej utraty danych przy zniszczeniu nośnika danych w przypadku taśm. Jest to także proces powolny, co w efekcie może spowodować poniesienie większych kosztów niż wartość okupu.

Rozwiązaniem, które wydaje się być aktualnie najlepszym, jest możliwie jak najwcześniejsze wykrycie potencjalnego źródła ataku. W przypadku, gdy te czynności zawiodą, jedyną możliwością na zmniejszenie strat jest minimalizacja skutków ataku na bieżąco. Aby tego dokonać, konieczne jest wczesne wykrycie ataku.

⁶ *Cost of a Data Breach Report 2022*, figure 1, s. 9.

⁷ *Check Point Research: Weekly Cyber Attacks increased by 32% Year-Over-Year; 1 out of 40 organizations impacted by Ransomware*, figure 1.

Rysunek 5. Incydynty ransomware per sektor gospodarki⁸.

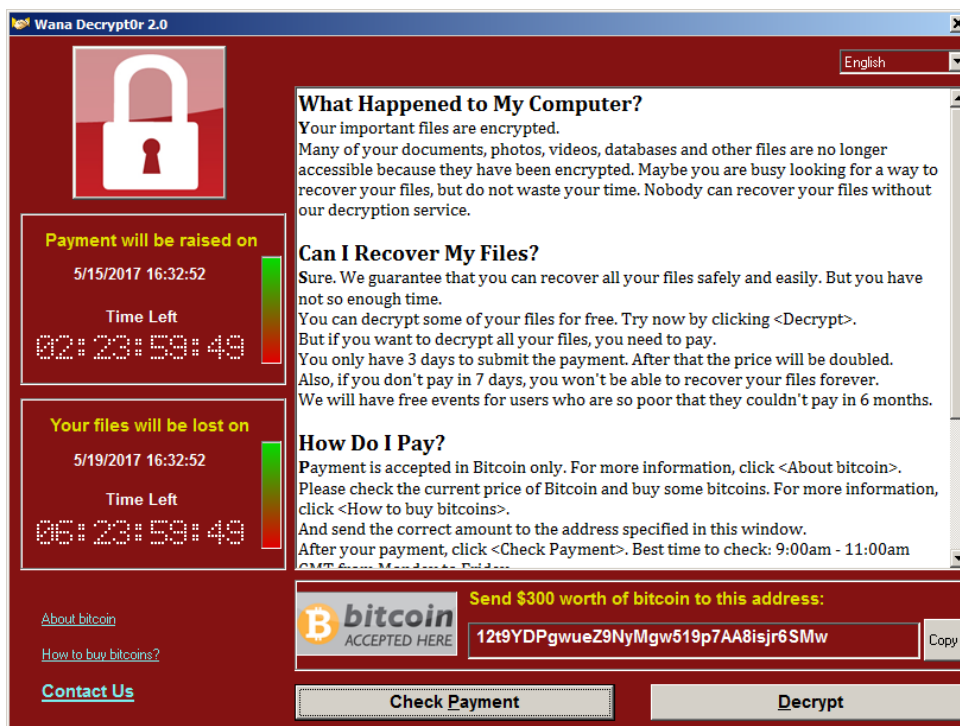
1.3 Krótka charakterystyka ataków ransomware

Ransomware można zdefiniować jako oprogramowanie, które blokuje atakowanemu dostęp do danych, do momentu zapłacenia okupu [8]. Prostsze ataki mogą sprowadzać się do blokady systemu bez uszkodzania plików, jednak większym zagrożeniem są tzw. „cryptovirological attacks” [9], czyli ataki wykorzystujące szyfrowanie danych jako formę blokady danych. Atakowany, jeśli nie posiada kopii zaszyfrowanych danych, musiałby odnaleźć klucz, którego użyto w szyfrowaniu. Nawet jeśli atakowany wie jakiego algorytmu użyto w ataku, to odnalezienie klucza jest problemem trudnym, zwłaszcza dla nowoczesnych algorytmów szyfrowania. Przykładowo, algorytm „AES” w zależności od klucza występuje w wariantach 128,192 oraz 256-bitowych, co daje między 2^{128} a 2^{256} możliwych wartości do sprawdzenia atakiem siłowym.

Przy wyłudzeniu okupu, atakujący stosują również techniki zastraszenia. Przykładowo wirus „WannaCry”, którego duża fala ataków miała miejsce w 2017 roku [10], informował, że początkowy

⁸Dragos Industrial Ransomware Attack Analysis: Q2 2023, figure 2.

okup 300\$ per maszyna wzrośnie dwukrotnie po 3 dobach zwłoki. Po upływie tygodnia odzyskanie danych miałyby stać się niemożliwe. Atakujący wymagają, aby okup został spłacony w sposób trudny do wyśledzenia przez organy ścigania m.in. za pomocą kryptowalut.



Rysunek 6. Ekran wyświetlający się po zainfekowaniu komputera przez WannaCry. Atakujący wymaga od ofiary zapłaty Bitcoinem⁹.

Ataki ransomware, mogą także założyć blokadę powłoki systemowej lub nawet dokonać modyfikacji partycji rozruchu jak w przypadku wirusa RedBoot [11].



Rysunek 7. Ekran rozruchu przy infekcji wirusem RedBoot¹⁰.

⁹<https://www.galsys.co.uk/news/wp-content/uploads/WannaCry-Pop-Up.jpg>

¹⁰<https://www.bleepstatic.com/images/news/ransomware/r/redboot/header.png>

Konceptualnie „cryptovirological attack” został przedstawiony w 1996 roku na konferencji IEEE Security & Privacy [12]. Opisuje się go jako protokół pomiędzy atakowanym, a atakującym:



Rysunek 8. Diagram sekwencji ataku ransomware.

Generowany klucz symetryczny ma charakter losowy i nie pomoże w odszyfrowaniu danych innej ofiary. Klucz prywatny jest przechowywany wyłącznie przez atakującego. Jedyne kontakty, jakie musi być wykonane bezpośrednio przez atakującego, następują w momencie, kiedy zaszyfrowany klucz symetryczny jest wysyłany do atakującego, a następnie klucz odszyfrowany do atakowanego.

Typowymi sposobami propagacji ransomware są:

- podszywanie się pod znane aplikacje czy strony internetowe,
- skuszenie ofiary do otworzenia niezauważanego załącznika listu elektronicznego,
- luki bezpieczeństwa sieci.

Rozdział 2

Przegląd literatury

W artykule opublikowanym przez firmę Microsoft¹ o tytule „Co to jest cyberbezpieczeństwo?”, trzy z sześciu wymienionych typów zagrożenia to:

- oprogramowanie wymuszające okup,
- inżynieria społeczna,
- wyłudzenie informacji.

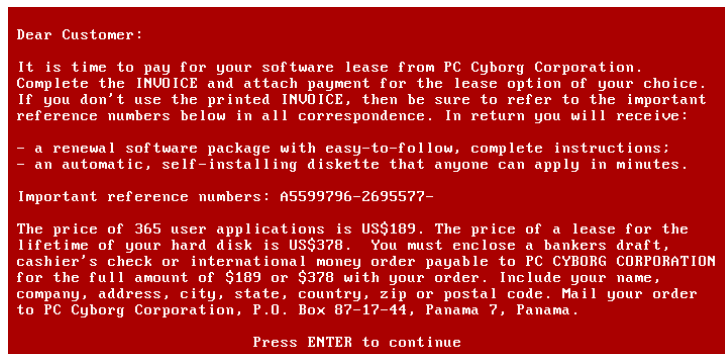
Atak ransomware zawiera w sobie każde z tych zagrożeń. Oprogramowanie złośliwe wymaga od ofiary zaufania, że to co uruchamia jest nieszkodliwe. Typowo propagacja takiego malware ma miejsce poprzez tzw. „phishing” czyli podszywanie się atakującego za zaufany serwis lub instytucję z którymi ofiara mogła wejść w interakcję w przeszłości. Aby zrozumieć zakres tych technik oraz możliwe wektory ataku, należy prześledzić ich historię.

2.1 Historia i ewolucja ataków typu ransomware

2.1.1 Wczesna historia

Mimo stopniowego nasilania się ataków ransomware w przeciągu ostatnich 7 lat sama idea utrudnienia dostępu do plików pod groźbą okupu jest znana od dosyć dawna. Już w drugiej połowie lat 80-tych, w USA, cyberprzestępcy w zamian za odzyskanie dostępu do danych wyłudжали okup, który następnie był wysyłany drogą pocztową. Jednym z pierwszych udokumentowanych ataków wirusem ransomware był DOSowy „AIDS trojan” [13] z 1989 roku. Autor programu — Joseph Popp — przekazywał dyskietki drogą pocztową do wybranej grupy ofiar pod przykrywką załącznika do ulotki informacyjnej na temat wirusa AIDS. Program modyfikował plik AUTOEXEC.BAT, z którego korzystał w celu zliczenia ilości uruchomień komputera. W momencie przekroczenia liczby 90 uruchomień szyfrował nazwy wszystkich plików na dysku C:, tym samym uniemożliwiając korzystanie z systemu.

¹Artykuł jest dostępny pod adresem: <https://www.microsoft.com/pl-pl/security/business/security-101/what-is-cybersecurity>

A screenshot of a ransomware message displayed in a black terminal window with white text. The message is from PC Cyborg Corporation and demands payment for a software lease. It lists important reference numbers and provides instructions on how to pay, including options for cash, check, or money order. The message ends with a prompt to press ENTER to continue.

```
Dear Customer:

It is time to pay for your software lease from PC Cyborg Corporation.
Complete the INVOICE and attach payment for the lease option of your choice.
If you don't use the printed INVOICE, then be sure to refer to the important
reference numbers below in all correspondence. In return you will receive:

- a renewal software package with easy-to-follow, complete instructions;
- an automatic, self-installing diskette that anyone can apply in minutes.

Important reference numbers: A5599796-2695577-

The price of 365 user applications is US$189. The price of a lease for the
lifetime of your hard disk is US$378. You must enclose a bankers draft,
cashier's check or international money order payable to PC CYBORG CORPORATION
for the full amount of $189 or $378 with your order. Include your name,
company, address, city, state, country, zip or postal code. Mail your order
to PC Cyborg Corporation, P.O. Box 87-17-44, Panama 7, Panama.

Press ENTER to continue
```

Rysunek 9. Wiadomość ukazująca się po aktywacji wirusa „AIDS trojan”²

Atakujący podszywał się pod fikcyjną korporację „PC Cyborg Corporation”, na której adres w Panamie miał być wysyłany okup. Paczka razem z dyskietką posiadała również ulotkę z krótkim wprowadzeniem, instrukcją obsługi, a także licencją co było w tamtym czasie powszechną i budzącą zaufanie praktyką. Program nie szyfrował treści samych plików, jedynie ich nazwy. Klucz szyfrowania był kluczem symetrycznym, co sprawiało, że złamanie go mogło pomóc odblokować system, każdej ofierze borykającej się z tą samą wersją wirusa. Eliminacja tej wady była inspiracją dla pracy „Cryptovirology: Extortion-Based Security Threats and Countermeasures”, w której przedstawiono pojęcie „cryptovirological attack” [12].

Po roku 1996, w erze upowszechnienia się internetu, pojawiły się sporadyczne ataki ransomware na niewielką skalę, tym razem ulepszone o szyfrowanie hybrydowe. W latach dwutysięcznych pojawił się trudny do wykrycia „PGPCoder” [14] używający 660-bitowego klucza RSA. Innym ransomware występującym w tamtym czasie był „Archivus” [15], również używający klucza RSA, w wersji 1024-bitowej, którego tragiczną wadą było używanie tego samego klucza do szyfrowania każdego pliku na każdej zainfekowanej maszynie. Ataki te, aby zainfekować ofiarę, wykorzystywały phishing i podszywały się pod zaufane strony internetowe.

2.1.2 Historia współczesna

Mimo historii sięgającej jeszcze lat 80 - tych, ataki ransomware nie były szczególnie powszechne w latach dwutysięcznych. Status quo został zachwiany po upowszechnieniu się kryptowalut, umożliwiających poufną i trudną do wyśledzenia wymianę środków między ofiarą a atakującym. Jednak uzyskanie pieniędzy od ofiar niezaznajomionych z kryptowalutami nie było proste, dopiero kantory kryptowalut dały cyberprzestępcom możliwość prostego i poufnego wyłudzenia środków. Pierwsza dekada XXI w. była dla cyberprzestępców czasem udoskonalania „scareware”, czyli oprogramowania mającego wystraszyć ofiarę na tyle, żeby zapłaciła za odzyskanie dostępu do stacji, bez wyrządzania szczególnej szkody na danych.

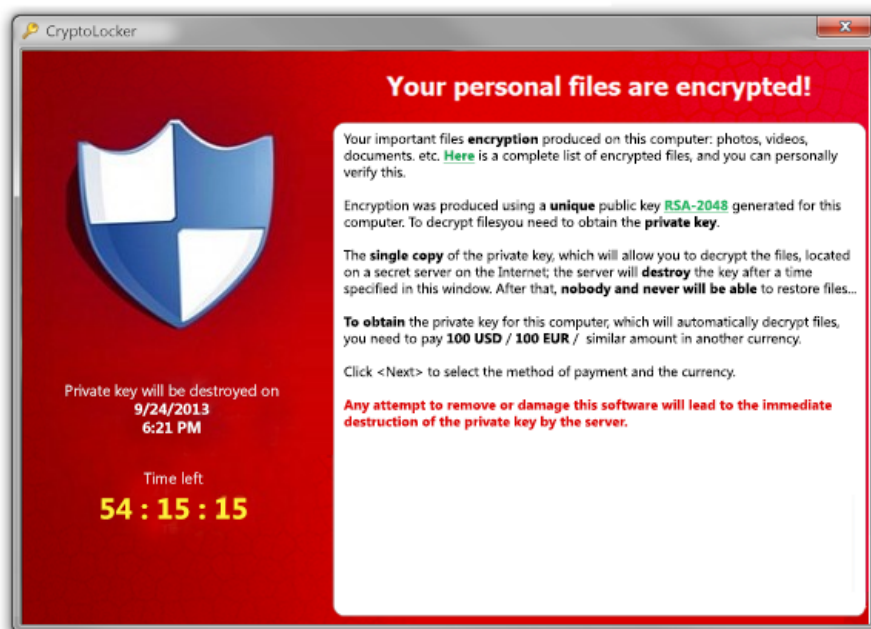
W 2013 roku w annały historii internetu wszedł Windowsowy wirus „CryptoLocker”. Wykorzystywał on do szyfrowania 2048-bitową parę kluczy RSA, generowaną na osobnym serwerze, a następnie

²<https://sophosnews.files.wordpress.com/2012/09/aids-info-demand-500.png>

dostarczał klucz publiczny na stację ofiary w celu szyfrowania jej plików [16]. Tym samym ofiara nie miała innej możliwości odzyskania plików niż zapłacić okup wynoszący 300 USD. Wirus dostarczany był jako załącznik w liście elektronicznym oraz przez owiany złą sławą „Gameover ZeuS botnet” [17]. Załącznik posiadał w sobie plik .zip, który z kolei zawierał w sobie plik .exe, z ikonką charakterystyczną dla pliku pdf. Atakujący wykorzystywał domyślne zachowanie Windowsa polegające na ukrywaniu rozszerzenia pliku. Następnie wirus podejmował następujące kroki:

1. rozpakowywał swoje pliki w ścieżce profilu użytkownika,
2. dodawał nową pozycję do windowsowego rejestru, który uruchamiał wirus wraz z rozruchem systemu,
3. pobierał klucz publiczny z jednego z serwerów,
4. wirus inicjował szyfrowanie plików na zamontowanych dyskach, w tym na dyskach sieciowych,
5. wyświetla ekran informujący o zdarzeniu i możliwości opłacenia okupu w BTC do 100 godzin od zaszyfrowania.

Po opłaceniu okupu ofiara miała możliwość pobrania programu dekodującego z załadowanym, odpowiednim kluczem prywatnym. Wirus szyfrował jedynie pliki z odpowiednimi rozszerzeniami m.in. pliki AutoCAD czy dokumenty MS Office.



Rysunek 10. Ekran wyświetlający się po zainfekowaniu komputera przez CryptoLocker³.

³<https://grzegorzkowalik.com/wp-content/uploads/2015/05/cryptolocker.png>

Zagrożenie tym wirusem zostało zneutralizowane w wyniku zainicjowanej przez departament sprawiedliwości USA, operacji „Tovar” [18] w wyniku której udało się uzyskać dostęp do bazy danych zawierającej prywatne klucze RSA na podstawie których możliwe było odzyskanie plików.

„CryptoLocker” był swego rodzaju kamieniem milowym w rozwoju cyberprzestępczości. Złożona natura procederu stała się normą dla ataków ransomware a wraz z coraz większą popularnością kryptowalut i usprawnionymi algorytmami szyfrowania asymetrycznego, ilość ataków oraz generowane przez nie straty stabilnie wzrastają aż do dnia dzisiejszego.

Aktualnie cyberprzestępcy zmienili styl ataku ze skupiającego się na infekcji jak największej ilości stacji, na tzw. „big game hunting” (BGH)⁴. w dużej mierze polega na koordynacji inżynierii społecznej i zaprojektowania oprogramowania ransomware w sposób, który będzie najbardziej szkodliwy dla dużych organizacji. Obierana jest mniejsza ilość celów na rzecz wyższej kwoty okupu. Raport „CrowdStrike Services” z 2023 roku donosi, że jedną najszerzej stosowanych taktyk BGH jest połączenie ransomware z groźbą upublicznienia skradzionych danych. Typowo dane zostają upublicznione gdy minie termin zapłaty okupu. Naruszenie danych jest rozłożone w czasie i wykorzystuje narzędzia już dostępne na atakowanym środowisku. Dzięki temu ataki są cięższe do wykrycia⁵. Techniki zastraszenia zostały także dopracowane, aby wywołać możliwie na największą presję na ofiarach. W przypadku „REvil” kradzione dane bywały etapowo upubliczniane, aby zmusić ofiarę do szybszego działania [19].

Z powodu dużej opłacalności takich ataków utworzony został model „ransomware as a service” (RaaS), w którym klienci płacą za dokonanie ataku ransomware programem utworzonym przez inne grupy hakerskie⁶.

Jednym z nich jest wcześniej wymieniony „REvil” używany przez grupę „PINCHY SPIDER”, którego cechą rozpoznawczą jest postowanie skradzionych danych na blogu „Happy Blog” [19]. W 2021 roku użyto go na wysoką skalę [20] przez podatność Kaseya VSA⁷ o identyfikatorze CVE-2021-30116 [21]. Atak ten można podsumować w następujących krokach:

1. użycie komendy PowerShell do zakończenia procesów Windows Defender,
2. podstawienie pliku wykonywalnego do katalogu instalacyjnego Windowsa,
3. zgodnie z techniką „Living off the land” wirus pobierał pomocnicze pliki wykonywalne i maskował je nazwami typowymi dla plików pomocniczych Windowsa np. agent.exe,
4. pobrane pliki następnie były przenoszone do odpowiednich folderów w celu załadowania ich razem z plikiem wykonywalnym MsMpeng.exe techniką nazywaną „DLL sideloading”⁸

⁴Dokładniejszą definicję z przykładami można znaleźć pod adresem:

<https://www.malwarebytes.com/blog/news/2023/07/ransomware-making-big-money-through-big-game-hunting>

⁵Technika ta nosi nazwę „Living off the land”

⁶Wykorzystywane jest oprogramowanie utworzone przez inne osoby, podobnie jak w modelu Software as a Service.

⁷Kaseya VSA jest narzędziem do zarządzania infrastrukturą IT.

⁸DLL sideloading polega na załadowaniu pliku binarnego o innej treści niż oryginalna. Wykorzystuje się ją do aktywacji serwisów lub wykonywania procesów w sposób trudny do wykrycia przez użytkownika.

5. w momencie wywołania przez MsMpeng.exe serwisów, na które ma zależności, ładowany jest podłożony wcześniej plik .dll, a razem z nim rozpoczyna się szyfrowanie danych na maszynie,
6. na pulpicie tworzony jest plik z instrukcją tłumaczącą jak spłacić okup w BTC na stronie ukrytej za TORem [22].

```

.text:013E10FC 68 04 1C 3F 01    push offset Type           ; "SOFTIS"
.text:013E1101 6A 65            push 65h                   ; lpName
.text:013E1103 6A 00            push 0                      ; hModule
.text:013E1105 FF D6            call esi ; FindResourceW
.text:013E1107 85 C0            test eax, eax
.text:013E1109 0F 84 98 00 00 00  jz loc_13E11A7
.text:013E110F 50              push eax                    ; hResInfo
.text:013E1110 6A 00            push 0                      ; hModule
.text:013E1112 FF 15 20 D0 3E 01    call ds:LoadResource
.text:013E1118 85 C0            test eax, eax
.text:013E111A 0F 84 87 00 00 00  jz loc_13E11A7
.text:013E1120 50              push eax                    ; hResData
.text:013E1121 FF 15 18 D0 3E 01    call ds:LockResource
.text:013E1127 68 14 1C 3F 01    push offset aModlis        ; "MODLIS"
.text:013E112C 6A 66            push 66h                   ; lpName
.text:013E112E 6A 00            push 0                      ; hModule
.text:013E1130 A3 40 43 3F 01    mov dword_13F43A0, eax
.text:013E1135 FF D6            call esi ; FindResourceW
.text:013E1137 85 C0            test eax, eax
.text:013E1139 0F 84 74 6C      jz short loc_13E11A7
.text:013E113B 50              push eax                    ; hResInfo
.text:013E113C 33 F6            xor esi, esi                ; hModule
.text:013E113E 56              push esi                    ; hModule
.text:013E113F FF 15 20 D0 3E 01    call ds:LoadResource
.text:013E1145 85 C0            test eax, eax
.text:013E1147 0F 84 74 5E      jz short loc_13E11A7
.text:013E1149 50              push eax                    ; hResData
.text:013E114A FF 15 18 D0 3E 01    call ds:LockResource
.text:013E1150 68 24 1C 3F 01    push offset aMpsvcDll      ; "mpsvc.dll"
.text:013E1155 BA 88 55 0C 00  mov edx, 0C5508h
.text:013E115A A3 44 43 3F 01    mov dword_13F43A4, eax
.text:013E115F 8B C8            mov ecx, eax
.text:013E1161 E8 9A FE FF FF    call Write_File_In_windows_folder
.text:013E1166 8B 0D A0 43 3F 01    mov ecx, dword_13F43A0
.text:013E116C BA D0 56 00 00 00  mov edx, 56D0h
.text:013E1171 C7 04 24 38 1C 3F 01    mov [esp+8+lpProcessInformation], offset aMmpengExe ; "MsMpEng.exe"
.text:013E1178 8B 03 FE FF FF    call Write_File_In_windows_folder
.text:013E117D C7 04 24 EC 43 3F 01    mov [esp+8+lpProcessInformation], offset ProcessInformation ; lpProcessInformation
.text:013E1184 68 A8 43 3F 01    push offset StartupInfo    ; lpStartupInfo
.text:013E1189 56              push esi                    ; lpCurrentDirectory
.text:013E118A 56              push esi                    ; lpEnvironment
.text:013E118B 68 30 02 00 00 00  push 230h                  ; dwCreationFlags
.text:013E1190 56              push esi                    ; bInheritHandles
.text:013E1191 56              push esi                    ; lpThreadAttributes
.text:013E1192 56              push esi                    ; lpProcessAttributes
.text:013E1193 FF 75 10          push [ebp+lpCommandLine]   ; lpCommandLine
.text:013E1196 C7 05 A8 43 3F 01 44 00 mov StartupInfo.cb, 44h
.text:013E11A0 50              push eax                    ; lpApplicationName
.text:013E11A1 FF 15 28 D0 3E 01    call ds:CreateProcessW

```

Rysunek 11. Miejsce w pliku binarnym agent.exe, w którym wywoływany jest MsMpeng oraz ładowany plik .dll⁹.

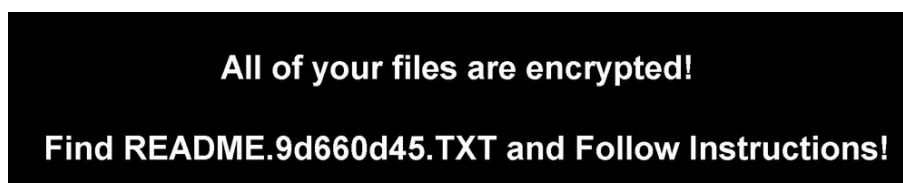
Innym znanym RaaS jest „DarkSide” używany przez grupę „CARBON SPIDER”. Do niedawna skupiał się głównie na atakach maszyn Windowsowych, niedawno rozszerzając się na systemy Linux, VMware ESXi i vCenter [23]. Wirus w wersji Windowsowej obchodzi zabezpieczenia kontroli użytkownika za pomocą interfejsu CMSTPLUA COM¹⁰, następnie sprawdza na podstawie lokalizacji i języka systemu w celu ominięcia ataku na maszynę z jednej z byłych republik radzieckich. Program podejmuje potem następujące kroki:

1. tworzy plik LOG.<id użytkownika>.TXT w którym przechowuje dane tymczasowe na temat progresu ataku,
2. usuwa pliki w koszu, programy antywirusowe i zapewniające bezpieczeństwo oraz zamyka procesy blokujące mu dostęp do danych użytkownika,
3. rozpoczyna szyfrowanie algorytmem Salsa20 przy pomocy losowo wygenerowanego klucza macierzowego,

⁹https://ik.imagekit.io/qualys/wp-content/uploads/2021/07/Fig.-5-Write_resource_Create_process.png

¹⁰Takie obejście można dokonać programem <https://github.com/tijme/cmstplua-uac-bypass>

4. klucz macierzowy jest szyfrowany zakodowanym na twardo kluczem RSA, a następnie łączony z zaszyfrowanym plikiem,
5. pozostawia plik README.<id użytkownika>.TXT w którym wskazuje stronę ukrytą za TORem, na której ofiara ma dokonać płatność w BTC lub XMR.



Rysunek 12. W wyniku działania wirusa tapeta użytkownika zostaje zmieniona na taką, jak widać na obrazku¹¹.

2.2 Istniejące techniki wykrywania i obrony przed ransomware

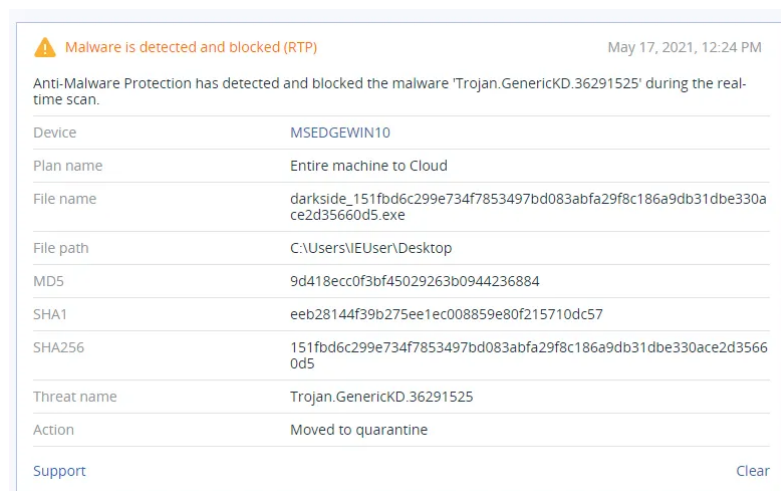
Niezależnie od tego czy atakujący korzysta z techniki „living off the land” lub stara się spowodować starty w możliwie najmniejszym przedziale czasowym, kluczem w minimalizacji kosztów ataku ransomware najważniejsza jest szybka reakcja. Aby to osiągnąć należy podjąć inteligentną strategię, która doprowadzi do możliwie jak najwcześniejszego wykrycia ataku. Jeśli administrator zostanie poinformowany dostatecznie wcześniej o zagrożeniu, będzie możliwa izolacja, a następnie eliminacja zagrożenia. Takie podejście w połączeniu ze zdyscyplinowanym harmonogramem kopii zapasowych, jest w stanie zredukować straty niemalże do zera.

Wyróżnia się trzy główne metody wykrywania ataku: poprzez sygnaturę plików, poprzez analizę nietypowego dla systemu zachowania oraz poprzez monitorowanie ruchu sieciowego [24].

2.2.1 Wykrywanie poprzez sygnaturę plików

Zasada działania tego typu wykrywania jest bardzo prosta. Oprogramowanie ma pewne unikalne cechy, na podstawie których wyliczana jest jego sygnatura. Do tych cech należą zakodowane na twardo nazwy domen, adresy IP oraz inne identyfikatory. Typowo także wykorzystywana jest wartość funkcji skrótu. Aby metoda ta mogła być skuteczna musi istnieć często aktualizowana baza danych zawierająca sygnatury wszystkich napotkanych typów ransomware. Niestety sposób ten jest ograniczony do wirusów napotkanych w przeszłości i nie jest nim możliwe wykrycie unikalnego zagrożenia.

¹¹<https://staticfiles.acronis.com/images/content/5cd67c66ec1401b8e67aee9e1bb04cc4.webp>



Rysunek 13. Tradycyjne antywirusy tak jak pokazany na obrazku Acronis, korzystają z metody wykrywania poprzez sygnaturę¹².

2.2.2 Wykrywanie poprzez analizę zachowania systemu

W przeciwieństwie do wcześniej wymienionego sposobu, wykrywanie poprzez analizę zachowania systemu nie opiera się na sprawdzeniu treści pliku wykonywalnego, a na wykryciu kroków, charakterystycznych dla naruszenia bezpieczeństwa systemu. W przeciwieństwie do poprzedniego rozwiązania, ta metoda jest przystosowana do kontrowania techniki „living off the land”. Dziedzina wykrywania behawioralnego wirusów stała się m.in. obiektem badań algorytmami opartymi o sztuczną inteligencję [24]. Branych pod uwagę może być wiele zdarzeń, z których najbardziej charakterystyczne są:

- wywoływanie pewnej grupy komend powłoki systemu,
- pobieranie otwarto-źródłowych programów do penetracji systemów,
- wykorzystywanie pewnej grupy zmiennych środowiskowych jako argumenty wywołań,
- użycie pewnej grupy wywołań systemowych w ciągu, jedno po drugim,
- duży ruch w katalogach domowych użytkowników lub w /tmp,
- zmiana atrybutów i właścicieli plików, katalogów czy punktów montowania dysków.

Jedną z najpowszechniejszych metod, używaną przez atakujących do powiadomienia ofiary o ataku i metodzie odzyskania dostępu do danych jest pozostawienie pliku tekstowego w miejscu łatwym do znalezienia np. w katalogu domowym użytkownika. Inną jest tworzenie pliku tymczasowego przechowującego stan zaawansowania ataku. Ze względu na to, część metod wykrywania ransomware skupia się na wyszukiwaniu tego typu plików, na podstawie treści techniką „bag-of-words”¹³ w celu odnalezienia korelacji między terminami typowymi dla takich dokumentów np. „encrypted”, „ransom” etc.

¹²<https://staticfiles.acronis.com/images/content/c110e0139779aec495bb2bd6e96ee4cd.webp>

¹³ Jest to technika przedstawienia tekstu w modelu nieułożonej kolekcji słów. Wykorzystuje się ją m.in. w przetwarzaniu języka naturalnego.

2.2.3 Wykrywanie poprzez analizę ruchu sieciowego

Wykrywanie poprzez analizę ruchu sieciowego polega na ograniczeniu analizy behawioralnej do wyłącznie monitorowania adresatów i treści pakietów komunikacji sieciowej. Szczególne zainteresowanie stanowią transfery danych do maszyn o nieznanym i podejrzanym adresach oraz domenach. Zgodnie z techniką „living off the land”, atakujący stara się możliwie minimalizować komunikację z serwerami zewnętrznymi, które mogą zostać uznane za podejrane. Mimo to znakomita większość narzędzi hakerskich, jest ogólnodostępna i dobrze znana w branży cyberbezpieczeństwa i tym samym łatwa do wykrycia [25].

Narzędzie	Strona
7zip	7-zip.org
AdFind	joeware.net
Advanced IP Scanner	advanced-ip-scanner.com
AnyDesk	anydesk.com
Proces Hacker	processhacker.sourceforge.io
rclone	rclone.org
WinSCP	winscp.net

Tabela 1. Tabela popularnych narzędzi wykorzystywanych w atakach ransomware¹⁴.

2.3 Podstawy działania systemów plików

Struktura i działanie systemu plików na Linuksach jest bardzo szerokim tematem. Na potrzeby analizy behawioralnej ataku ransomware przybliżę w tej sekcji po krótku działanie i wybrane, interesujące szczegóły implementacyjne.

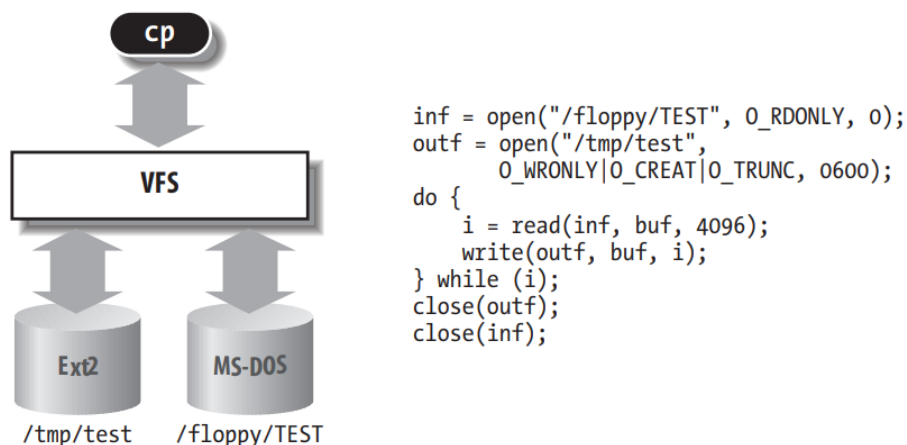
2.3.1 Skrócony opis działania systemu plików

Najpopularniejsze dystrybucje systemu Linux korzystają w większości z systemu plików o nazwie ext4. Wprowadzony do repozytorium jądra systemowego w 2008 roku, zyskał wielkie poważanie dzięki nowoczesnej obsłudze nośników danych oraz ulepszeniu systemu księgowania operacji, wprowadzanego w ext3. Księgowanie operacji w systemie plików polega na przechowywaniu zapisów jako *transakcji*. Dopiero jeśli transakcja zakończy zapisywanie na dysk, jej dane zostają wprowadzone na system plików [26]. W efekcie oznacza to, że w wypadku zaniechania działania systemu w trakcie zapisu, transakcja zostanie cofnięta po ponownym rozruchu i tym samym zachowana zostanie spójność systemu plików.

Obsługa wielu rodzajów systemu plików jest możliwa dzięki istnieniu *virtualnego systemu plików*. Jest on warstwą abstrakcji pomiędzy konkretnymi jej implementacjami, a aplikacjami klienckimi. Dzięki

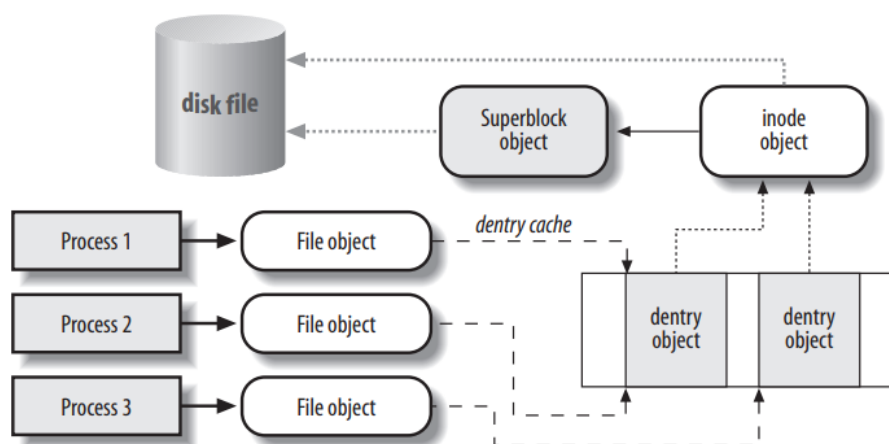
¹⁴Dane pochodzą ze strony: <https://lots-project.com/>

temu możliwa jest spójna i ujednolicona interakcja z systemem plików oraz interoperowalność między różnymi jego implementacjami [27].



Rysunek 14. Rola wirtualnego systemu plików w operacji kopiowania¹⁵.

Ogólna zasada działania wirtualnego systemu plików polega na podmienianiu przez nią typowych wywołań systemowych takich jak `read` lub `write` na funkcje natywne dla konkretnego systemu plików np. ZFS lub wcześniej wymieniony EXT4. Każda implementacja musi móc przetłumaczyć swoją wewnętrzną strukturę organizacyjną na model ogólny wirtualnego systemu plików [27].



Rysunek 15. Interakcja pomiędzy procesami a obiektami wirtualnego systemu plików¹⁶.

Informacje na temat interakcji pomiędzy otwartym plikiem a procesem są przechowywane w charakterystycznym dla procesu otwierającego plik „file object”. Informacje te istnieją *wyłącznie* w pamięci jądra systemu kiedy plik jest otwarty przez proces.

¹⁵ *Understanding Linux Kernel 3rd edition*, Figure 12-1 s. 457.

¹⁶ *Understanding Linux Kernel 3rd edition*, Figure 12-2 s. 460.

2.3.2 Monitorowanie zmian na systemie plików

Jądro systemu, nie może utrzymywać zakodowanej na twardo implementacji operacji na systemie plików ze względu na ich różnorodność. Utrzymywany jest więc indeks wskaźników do odpowiednich implementacji operacji. Taka struktura komunikacji między systemem plików a jądrem pozwala na śledzenie wywołań oprogramowaniem pośrednim. Jądro Linux zawiera w sobie dwie ciekawe z poziomu tematu pracy implementacje takich „pośredników”: „inotify subsystem” oraz „Linux Auditing Framework”.

API inotify

Podsystem inotify został stworzony z myślą o monitorowaniu oraz powiadamianiu o zmianach na dysku [28]. Jego głównym przypadkiem użycia jest automatyczne aktualizowanie widoków katalogów, plików konfiguracyjnych, zmian logów systemowych i tym podobnych. Rozwiązanie to znajduje się w kodzie źródłowym jądra Linux od sierpnia 2005 roku. Interfejs programowalny dla tego narzędzia zawiera się w bibliotece inotify-tools, które zawiera w sobie również pakiet narzędzi będących gotowymi implementacjami funkcjonalności API [29].

```
1  $ cat inotify-test.sh
2  #/bin/bash
3  inotifywait -m /home/user/box -e create -e moved_to |
4  while read -r directory action file; do
5      echo "File has been created!"
6  done
7  $ ./inotify-test.sh
8  Setting up watches.
9  Watches established.
10 [1] + 13180 suspended  ./inotify-test.sh
11 $ touch box/h2
12 $ fg
13 [1] + 13180 continued  ./inotify-test.sh
14 File has been created!
```

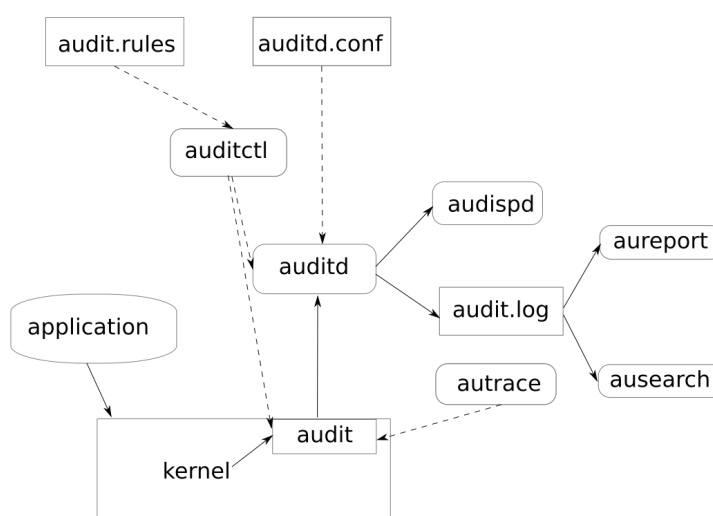
Listing 1. Przykład użycia narzędzia inotifywait. Po utworzeniu pliku ukazała się odpowiednia wiadomość

Niestety rozwiązanie to nie jest perfekcyjne i ma swoje limity. Do nich należą:

- brak wsparcia dla rekurencyjnego obserwowania ścieżek,
- „gubienie” niektórych wydarzeń dla starszych wersji jądra Linux,
- brak wsparcia niektórych wydarzeń przed wersją 5.13 jądra Linux [30],
- brak obserwacji dysków sieciowych.

Linux Auditing Framework

Projekt Linux Auditing Framework to podsystem wbudowany w jądro systemu Linux, którego zadaniem jest przechwytywanie, a następnie logowanie operacji systemowych. Jego możliwości nie ograniczają się wyłącznie do obserwacji systemu plików. Jest on w pełni zgodny z CAPP¹⁷, a więc może być używany jako wiarygodne źródło informacji o stanie systemu. Informacje można pobierać dzięki aplikacji po stronie użytkownika o nazwie `auditd`. Za pomocą komponentu `auditd` możliwe jest zapisanie logów do pliku lub wysłanie ich UNIXowym gniazdkiem do innych aplikacji.



Rysunek 16. Bardzo uproszczony diagram komponentów LAF¹⁸.

Niestety bardzo ciężko jest znaleźć informacje na temat implementacji części systemu która znajduje się w jądrze, ale na podstawie własnej analizy kodu zawartego w repozytorium głównym projektu¹⁹, w szczególności w plikach `audit.c`, `audit_fsnotify.c` oraz `auditd.c` w folderze `kernel`, mogę z dużą dozą pewności stwierdzić, że informacje wykryte tym narzędziem są wiarygodne i przydatne z perspektywy tematu pracy. W branży administracji systemami jest to narzędzie dobrze znane i poważane dzięki możliwościom łatwej i bezinwazyjnej konfiguracji. Popularne dystrybucje serwerowe takie jak Ubuntu Server, SLES, Red Hat oraz Fedora wspierają w pełni funkcjonalności związane z monitorowaniem systemu plików.

2.3.3 Krótka charakterystyka plików wykonywalnych

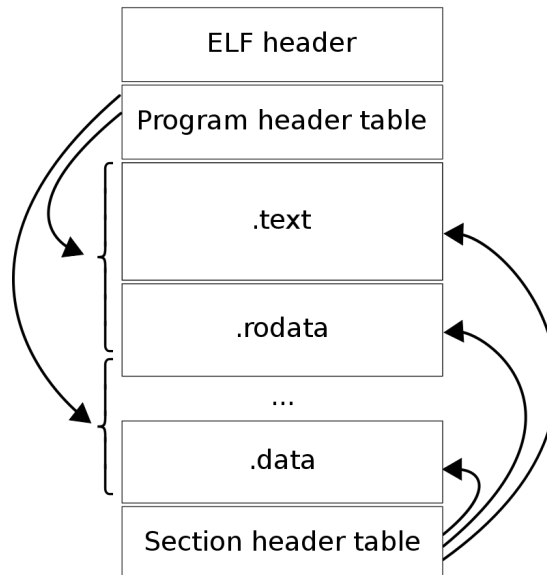
System Linux pliki wykonywalne zapisuje i odczytuje w formacie ELF czyli „Executable Linking Format” [31]. Najciekawszym elementem tego formatu, z punktu widzenia tej pracy, jest nagłówek. Mimo, że nie zawiera tak wielu informacji co nagłówki plików wykonywalnych na systemie Windows,

¹⁷Controlled Access Protection Profiles to środowisko służące do niezależnej oceny, analizy i testowania produktów w celu ustanowienia wymagań bezpieczeństwa.

¹⁸<https://documentation.suse.com/sles/12-SP5/html/SLES-all/cha-audit-comp.html>

¹⁹Pod adresem: <https://github.com/linux-audit/audit-kernel>

warto przyjrzeć się mu aby móc zidentyfikować obecność oprogramowania złośliwego lub narzędzia typowo wykorzystywanego podczas naruszenia bezpieczeństwa systemu.



Rysunek 17. Podział wewnętrzny pliku ELF²⁰.

W mojej opinii ciekawą sekcją jest `.note.gnu.build-id` [32]. Cytując `elf(5)` Linuksowego `man` pages: „This section is used to hold an ID that uniquely identifies the contents of the ELF image. Different files with the same build ID should contain the same executable content [...]”. Oznacza to, że można dzięki niemu *zidentyfikować konkretną kompilację aplikacji*. W przeciwieństwie do systemu Windows, gdzie typowo użytkownik pobiera już wcześniej przekompilowane pliki wykonywalne, bardzo popularnym rozwiązaniem na Linuksach jest kompilowanie lokalnie. Wyjątkiem są zaufane repozytoria, do których dostęp uzyskuje się przez menadżer pakietów dodawany do danej dystrybucji, np. `apt`. Mimo, że możliwa jest identyfikacja zawartości pliku binarnego poprzez wyliczenie jej wartości funkcji skrótu w niedługim czasie funkcją `md5`, identyfikator kompilacji dla tych samych warunków kompilacji i zawartości kodu wykonywalnego, będzie dokładnie taki sam. Informacja ta może być wykorzystywana do identyfikacji tego czy podejrzany plik binarny został skompilowany lokalnie z kodu źródłowego.

²⁰<https://upload.wikimedia.org/wikipedia/commons/7/77/Elf-layout--en.svg>

```
1 $ cargo build --release
2     Finished release [optimized] target(s) in 0.13s
3 $ readelf --notes target/release/linux-fs-audit | grep "Build ID"
4     Build ID: ff6019887a97bedc98a8eca3267817233a13a8bc
5 $ rm target/release/linux-fs-audit
6 $ cargo build --release
7     Finished release [optimized] target(s) in 0.04s
8 $ readelf --notes target/release/linux-fs-audit | grep "Build ID"
9     Build ID: ff6019887a97bedc98a8eca3267817233a13a8bc
```

Listing 2. Test rekompilacji aplikacji napisanej w języku Rust. Mimo ponownej kompilacji, przy braku zmiany kodu źródłowego, identyfikator pozostał ten sam. Można więc z dużą pewnością stwierdzić, że plik wykonywalny był skompilowany na tej maszynie, a nie pobrany z internetu.

2.4 Metody analizy statystyk systemu plików

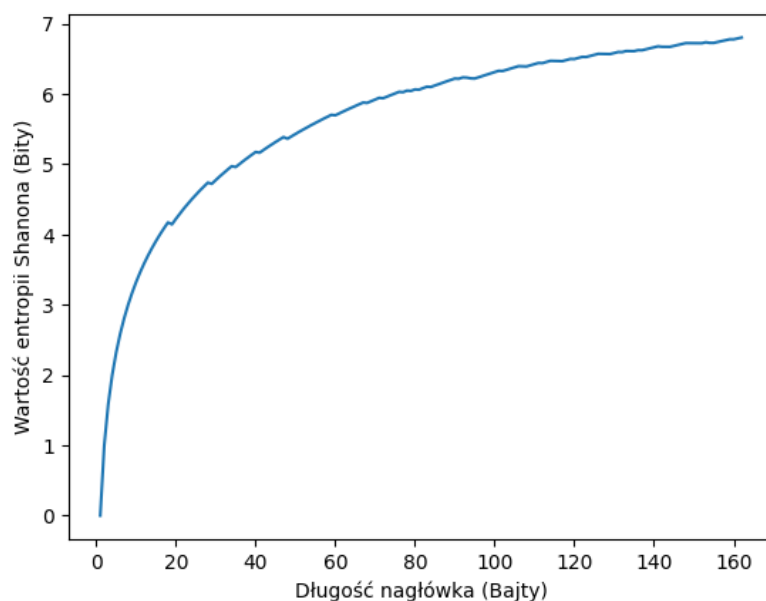
Wraz ze wzrostem ryzyka ataków ransomware, wzrosła też ilość prac opisujących możliwe metody wykrywania ataków na podstawie statystyk systemowych. W tym rozdziale chciałbym wymienić i po krótko wytłumaczyć, w mojej opinii, najciekawsze z nich.

2.4.1 Analiza entropii pliku

W pracy „Differential area analysis for ransomware attack detection within mixed file datasets” [33] przedstawiona jest metoda potencjalnego wykrycia tego czy plik został zaszyfrowany poprzez obliczenie entropii pliku dla różnych wielkości nagłówka. Nagłówek w kontekście tej metody po prostu oznacza ilość bajtów braną pod uwagę w obliczaniu entropii, a niekoniecznie twardo sprecyzowany w specyfikacji rodzaju pliku obszar na jego początku. Maksymalna możliwa entropia per bajt dla pliku jest równa ośmiu bitom na jeden bajt, wartość sugerująca kompletnie losową naturę pliku. Wzór na entropię H [34] to:

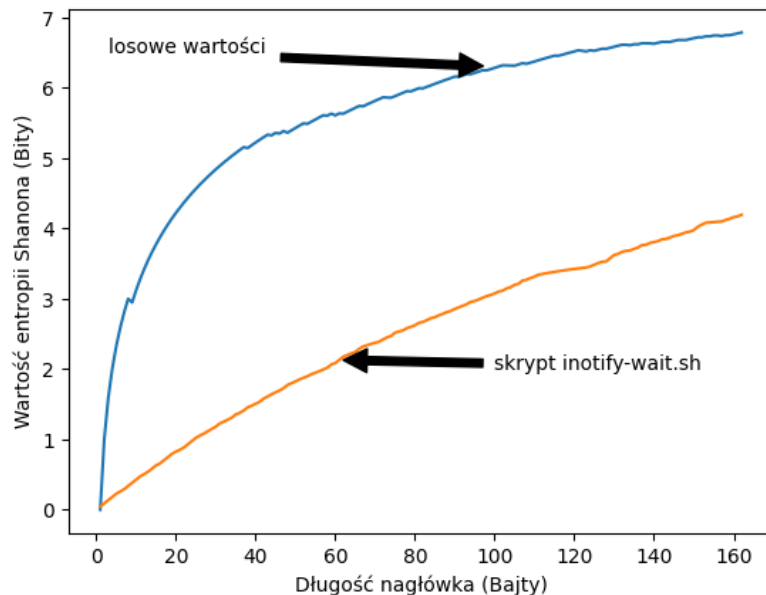
$$H(X) = - \sum_{i=1}^n P(x_i) \log_2 P(x_i)$$

gdzie n jest liczbą bajtów w próbce, a $P(x_i)$ to prawdopodobieństwo wystąpienia bajtu i w strumieniu bitów. Wyobraźmy sobie, że mamy 150 bajtowy plik który został wygenerowany losowo. Jego wykres entropi naliczonej od długości nagłówka x będzie przypominał funkcję $\log_2(x)$.



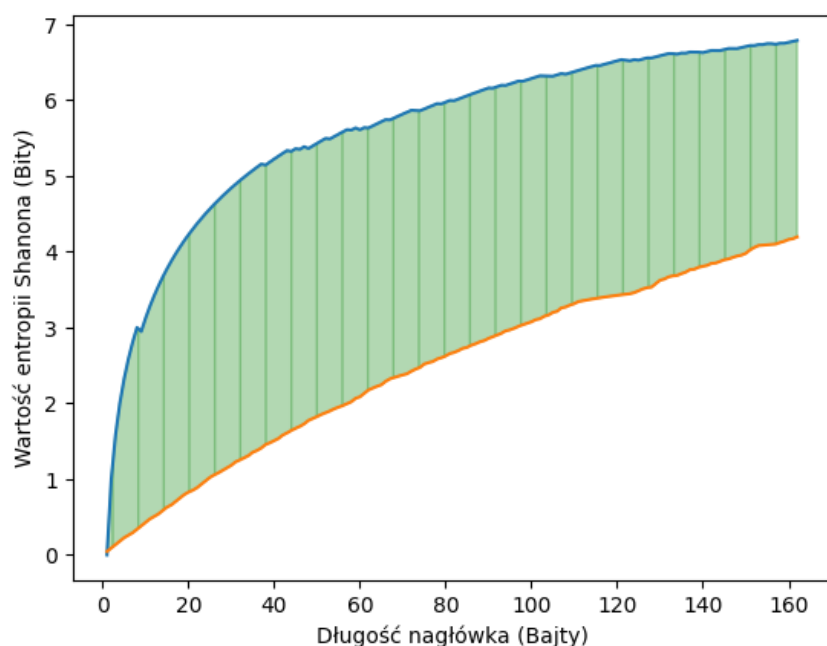
Rysunek 18. Wykres entropii od długości nagłówka dla pliku zawierającego zupełnie losowe dane.

Dla rzeczywistych plików wartość entropii będzie rosła w mniejszym tempie, ze względu na powtarzające się schematy w informacji.



Rysunek 19. Zestawienie wykresów entropii od długości nagłówka. Jako plik przykładowy wybrałem skrypt z sekcji Monitorowanie zmian na systemie plików.

Miarą tego jak duże jest prawdopodobieństwo, że plik został zaszyfrowany jest pole między tymi dwoma wykresami.



Rysunek 20. Pole między wykresem losowego i rzeczywistego pliku o takich samych długościach.

W pracy „Differential area analysis for ransomware attack detection within mixed file datasets” [33] zasugerowane są pewne wartości klasyfikacyjne, ustalone na podstawie dokładności wykrycia zaszyfrowanego pliku w zbiorach testowych.

Table 7 – Classification Accuracy Results (%).					
Header Length (Bytes)	Classification Criteria (Bit-Bytes)				
	8	24	40	56	72
32	63.290	20.097	19.439	14.061	11.767
64	99.010	87.851	46.472	24.227	19.879
96	98.546	99.914	99.842	87.077	51.326
128	98.118	99.903	99.944	99.794	89.289
160	96.736	99.825	99.960	99.934	96.439
192	97.452	99.744	99.957	99.952	98.959
224	97.234	99.631	99.954	99.957	99.517
256	97.055	99.505	99.944	99.962	99.713

Table 8 – Classification Precision Results (%).					
Header Length (Bytes)	Classification Criteria (Bit-Bytes)				
	24	40	56	72	
96	99.369	98.503	44.651	17.640	
128	100	99.462	98.058	49.324	
160	100	99.656	99.371	74.537	
192	100	100	99.553	90.923	
224	100	100	99.621	95.596	
256	100	100	99.690	97.343	

Rysunek 21. Skuteczność dla wybranych kryteriów klasyfikacji na długość nagłówka w bajtach²¹.

Metoda ta wydaje się być bardzo obiecująca lecz jest ograniczona wymogiem wielkości pliku. Jak widać skuteczność jest lepsza dla plików o rozmiarze większym niż 32 bajty.

²¹ *Differential area analysis for ransomware attack detection within mixed file datasets*, Table 7, Table 8, s. 11.

2.4.2 Automatyczna analiza behawioralna poprzez audyt systemu

W pracy „Automated Behavioral Analysis of Malware A Case Study of WannaCry Ransomware” [35] opisana jest metoda identyfikacji ransomware poprzez wyciągnięcie z logów audytu podczas rutynowego działania systemu. W przypadku tej pracy poszukiwanie abnormalnych zachowań systemu było oparte **wyłącznie** na wiedzy o tym, że atak ma miejsce. Aplikacja do audytowania systemu we wcześniej wymienionej pracy ma podobne możliwości do wymienionego w sekcji [Monitorowanie zmian na systemie plików](#) Linux Auditing Framework.

Przedstawiona w pracy metoda opera się o „Term-Frequency-Inverse-Document-Frequency (TF-IDF)” [36] czyli metrykę obliczania wagi słów w oparciu o liczbę wystąpień, dostosowaną do faktu generalnie częstszego występowania niektórych słów. Jest to metoda często wykorzystywana w jako forma wydobywania informacji z tekstów m.in. „text miningu”. TF-IDF jest produktem dwóch statystyk: częstości występowania słowa oraz odwrotnej częstości dokumentu. Częstość występowania słowa zapisuje się wzorem:

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$$

a odwrotną częstość dokumentu:

$$idf(t, D) = \log \frac{N}{1 + |d \in D : t \in d|}$$

gdzie słowo t , występujące w dokumencie d i wielkości zbioru dokumentów (corpus) N , występuje z częstością $f(t, d)$.

Niestety metoda ta nie przynosi szczególnych efektów. We wcześniej wymienionej pracy, zostały wykonane dwa eksperymenty: pierwszy w którym były wyłącznie logi z działania wirusa WannaCry oraz normalnych zachowań w systemie w osobnych dokumentach, drugi w którym przemieszane były działania wirusa z normalnym działaniem systemu w tym samym dokumencie.

Name	Meaning
b.wnry	Bitmap file for Desktop image
c.wnry	Configuration file
r.wnry	Q&A file, payment instructions
s.wnry	Tor client
t.wnry	WANACRY! file with RSA keys
u.wnry	@WannaDecryptor@.exe
\msg	Folder containing RTF files with payment instructions in 128 languages (e.g., korean.wnry)
taskse.exe	Launches decryption tool
taskdl.exe	Removes temporary files

Rysunek 22. Tabela plików tymczasowych wykorzystywanych przez wirus WannaCry²².

²²Automated Behavioral Analysis of Malware A Case Study of WannaCry Ransomware, Table I, s. 2.

Feature	Ranking (case 1)	Ranking (case 2)
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\s.wnry"	1	2
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\b.wnry"	2	7
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\u.wnry "	4	13
"enhanced:_object=file+event=read+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\t.wnry ", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_korean.wnry ", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_vietnamese.wnry "	7	32
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_chinese (traditional).wnry", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_japanese.wnry"	8	36
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_chinese (simplified).wnry", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_romanian.wnry "	9	40
"enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_bulgarian.wnry " ... (22 various language features) "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg\\m_turkish.wnry"	10	45
"enhanced:_object=file+event=read+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\c.wnry ", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\c.wnry", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\taskdl.exe", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\taskdl.exe", "enhanced:_object=registry+event=read+data=regkey: \\ activecomputernamemachineguid"	12	54
"enhanced:_object=registry+event=read+data=regkey: hkey_local_machine\\software\\microsoft\\cryptography\\defaults\\provider\\ microsoft enhanced rsa and aes cryptographic provider (prototype)image path"	9	58
"bigram:_api=regcreatekeyexw+arguments=software\\wanacrypt0r", "enhanced:_object=dir+event=create+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\msg", "enhanced:_object=file+event=execute+data=file:attrib +h . ", "enhanced:_object=file+event=execute+data=file:icacls . /grant everyone:f /t /c /q ", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\00000000.pky", "enhanced:_object=file+event=write+data=file: c:\\docume~1\\cuckoo\\locals~1\\temp\\r.wnry"	16	80

Rysunek 23. Tabela wyników z eksperymentu. Eksperyment pierwszy i drugi są tutaj nazwane „case 1” i „case 2”. Rankingi są wyliczone na podstawie wartości wagi TF-IDF ze wszystkich dokumentów²³.

Jak widać z tabelki, waga informacji o działaniu ransomware znacznie zmalała po połączeniu z logami działania systemu. Audyt systemu może być przydatny dla zaznajomionego z infrastrukturą administratora lecz sama jej analiza nie jest skuteczna w wykrywaniu ransomware. Tym samym informacja o dokonaniu operacji na systemie plików, sama w sobie nie wystarczy do wykrycia ataku.

²³ Automated Behavioral Analysis of Malware A Case Study of WannaCry Ransomware, Table V, s. 5.

2.4.3 Analiza podobieństwa pliku wykonywalnego

Praca „A Framework for Analyzing Ransomware using Machine Learning” [37] przedstawia jako daną do przetworzenia przez model sztucznej inteligencji podobieństwo cosinusowe wyliczone na podstawie treści wykonywalnego pliku binarnego. Podobieństwo cosinusowe jest metodą pomiaru podobieństwa pomiędzy dwoma niezerowymi wektorami długości n . Jego wartości zawierają się między zerem a jedynką. Jeśli dwa wektory mają taką samą orientację jego wartość wynosi jeden. Dla dwóch wektorów P oraz Q , podobieństwo będzie wynosić:

$$\cos(\theta) = \frac{P \cdot Q}{|P||Q|} = \frac{\sum_{i=1}^n P_i \cdot Q_i}{\sqrt{\sum_{i=1}^n P_i^2} \sqrt{\sum_{i=1}^n Q_i^2}}$$

W tym wypadku P i Q to dwa pliki wykonywalne, mogą to być wirusy lub zwyczajne programy codziennego użytku. Wektory te zostają utworzone na podstawie częstotliwości występowania instrukcji (z argumentami) kodu asemblera zdobytego z pliku binarnego, programem objdump. Przykładowo dla klasycznego programu:

```

1  #include <stdio.h>
2
3  int main() {
4      printf("Hello world!\n");
5      return 0;
6  }
```

Listing 3. Elementarny program w C.

do kodu asemblera będzie wyglądał w taki sposób:

```

1  000000000040104e <_start>:
2      40104e: f3 0f 1e fa          endbr64
3      401052: 66 90                xchg    %ax,%ax
4      401054: 31 ed                xor     %ebp,%ebp
5      401056: 49 89 d1             mov     %rdx,%r9
6      401059: 5e                  pop     %rsi
7      40105a: 48 89 e2             mov     %rsp,%rdx
8      40105d: 48 83 e4 f0          and     $0xffffffffffffffff,%rsp
9      401061: 50                  push    %rax
10     401062: 54                  push    %rsp
11     401063: 45 31 c0             xor     %r8d,%r8d
12     401066: 31 c9                xor     %ecx,%ecx
13     401068: 48 c7 c7 46 11 40 00 mov     $0x401146,%rdi
14     40106f: ff 15 53 2f 00 00    call    *0x2f53(%rip)
15     401075: f4                  hlt
16     401076: 66 2e 0f 1f 84 00 00 cs nopw 0x0(%rax,%rax,1)
17     40107d: 00 00 00
```

Listing 4. Fragment asemblerowego kodu programu z listingu 3.

Aby przedstawić jak wygląda ta metoda, spreparowałem plik zawierający wyłącznie instrukcje asemblerowe z argumentami. Z nich wyliczyłem częstość występowania poszczególnych słów, a następnie wyliczyłem współczynnik podobieństwa ze wzoru przedstawionego wcześniej. Dany jest program z listingu 3 oraz:

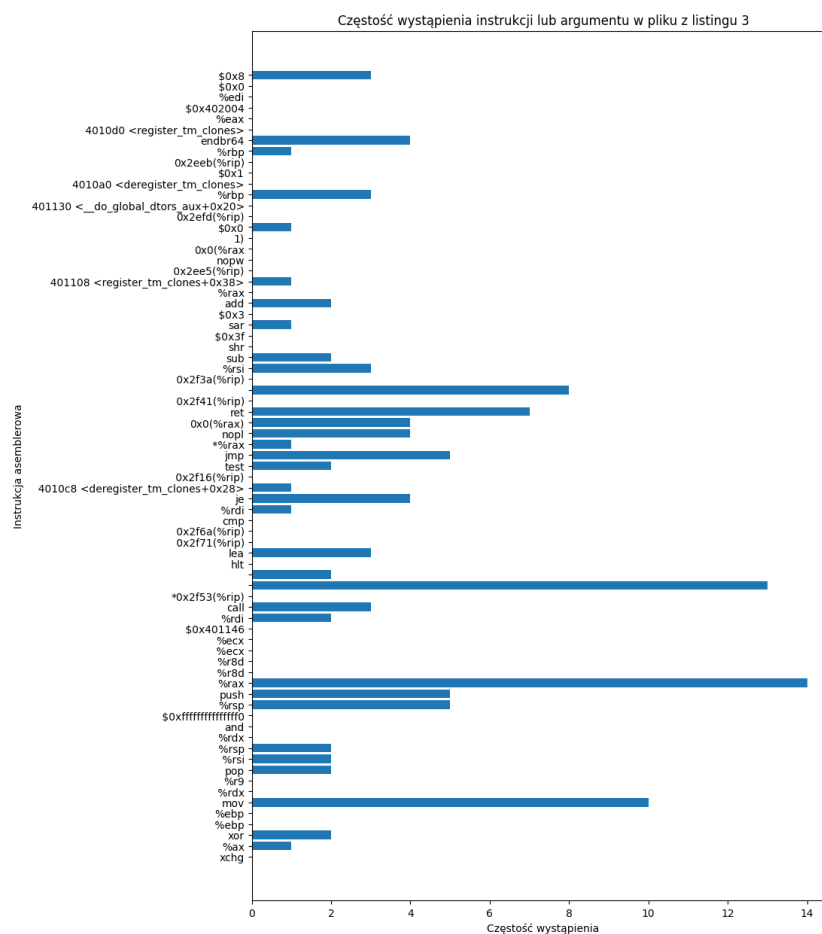
```

1      #include <stdio.h>
2
3      int main() {
4          int i = 2;
5          printf("Hello %d!\n",i);
6          return 0;
7      }

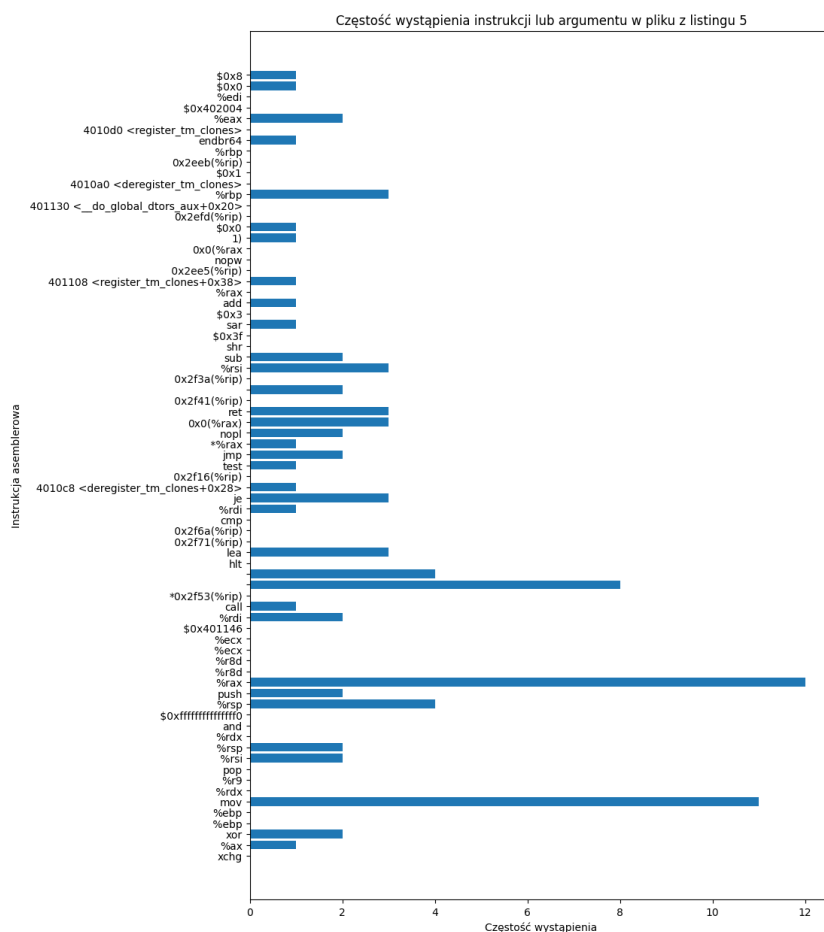
```

Listing 5. Program podobny do programu z listingu 3. Jedyną różnicą jest obecność zmiennej *i*.

Ich podobieństwo wyliczone ze wcześniej wymienionego wzoru wynosi $\cos(\theta) = 0.996289145765327$, czyli jest bardzo duże co pokrywa się z oczekiwaniami. Jego wartość *odzwierciedla* różnicę pomiędzy dwoma plikami, więc miara ta jest czuła na nawet niewielkie zmiany.



Rysunek 24. Prawdopodobieństwa wystąpienia bajtów o konkretnej wartości w pliku z listingu 3.



Rysunek 25. Prawdopodobieństwa wystąpienia bajtów o konkretnej wartości w pliku z listingu 5.

Jest to obiecująca metoda, która w przeciwieństwie do sprawdzania sygnatury pliku funkcją skrótu, adaptuje się do małych i średnich zmian w historycznie występujących zagrożeniach.

Rozdział 3

Analiza problemu

Bedąc uzbrojonym w informacje wymienione we wcześniejszych rozdziałach można ułożyć model działania programu będącego celem tej pracy. Aby to jednak było możliwe należy wybrać najważniejsze informacje o maszynie, systemie plików i odpowiednie strategie ich użycia.

3.1 Charakterystyka typowych zmian w systemie plików podczas ataku ransomware

Z informacji wymienionych w Historia i ewolucja ataków typu ransomware oraz w sekcji Istniejące techniki wykrywania i obrony przed ransomware można wyłuskać kilka punktów interakcji, które aktywują skan jedną z metod przedstawionych w Istniejące techniki wykrywania i obrony przed ransomware. Na potrzeby pracy proponuję:

1. przekroczenie obranej i dostatecznie wysokiej granicy wykonywanych operacji w ścieżce,
2. duża ilość usuniętych a potem dodanych po sobie plików w obranym oknie czasowym (sugerująca zmianę nazw),
3. powstanie plików zawierających w sobie słowa „README”, „LOG”, „ENCRYPTED”,
4. rutynowy skan w ustalonym przedziale czasowym.

Pierwszy i drugi punkt będzie wymagał od administratora dostosowania współczynników liczbowych i tym samym dokonania korekty na własną rękę. Powodem tej konfiguracji jest indywidualna natura ruchu na danej maszynie. Na niektórych ruch będzie większy niż na innych co jest warte wzięcia pod uwagę. Punkt trzeci mimo, że jest wskaźnikiem na pierwszy rzut oka prymitywnym, jest usprawiedliwiony zachowaniem dwóch najbardziej niebezpiecznych RaaS, wymienionych w sekcji Historia i ewolucja ataków typu ransomware. Rutynowy skan jest w dużej mierze ostateczną metodą która zostanie wykorzystana kiedy wszystkie inne zawiodą. Sama operacja skanowania będzie oparata na jednej lub kilku z metodach opisanych w sekcji Metody analizy statystyk systemu plików.

3.2 Wybór odpowiednich statystyk i metryk do analizy

Aby móc przeanalizować ruch na wycześniej wymienione sposoby potrzebne będą informacje o:

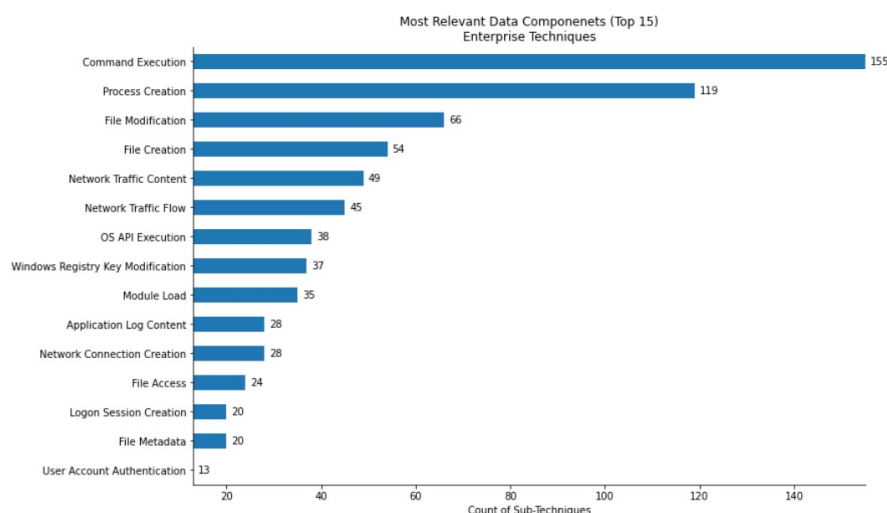
- czasie wystąpienia operacji w celu ustalenia ramki czasowej w której się zdarzyły,
- plikach zmodyfikowanych w ramach operacji,
- rodzaju operacji oraz plik wykonywalny który był jej źródłem,
- wywołaniu systemowym użytym w operacji.

Aby dokonać rzetelnego raportu dla administratora w moim przeczuciu należy także zawrzeć użytkownika oraz grupę będącą źródłem zaobserwowanej operacji. Zawarcie w tych statystykach nazwy wywołania systemowego ma znaczenie dla sprecyzowania jaka operacja *naprawdę* została dokonana [27]. Dany jest przykład użycia aplikacji Visual Studio Code:

```
1  $ pidof code
2  9551
3  $ strace -p 9551
4  strace: Process 9551 attached
5  restart_syscall(<... resuming interrupted read ...>
6  ) = 0
7  futex(0x7ffcab93e008, FUTEX_WAKE_PRIVATE, 1) = 0
8  lseek(26, 0, SEEK_SET) = 0
9  read(26, "296974470 53709 24257 29420 0 82"... , 4095) = 38
```

Listing 6. Przykładowo używając Visual Studio Code niektóre operacje mogą sprawiać pozory, że na pliku wywołano polecenie code bez wiedzy co dokładnie zaszło podczas działania programu.

W listingu wyżej można wydedukować na podstawie wiedzy o tym, że Visual Studio Code jest edytorem tekstowym, że otwarto plik z blokadą wykluczającą, wskaźnik w nim został przesunięty do miejsca zerowego dla pliku o deskryptorze numer dwadzieścia sześć oraz odczytano 4095 znaków od tamtego miejsca. Reasumując - odczytano plik do edycji i zablokowano do niego dostęp innym procesom.



Rysunek 26. Najbardziej wartościowe źródła danych o ataku wg. MITRE¹.

3.3 Potencjalne wyzwania i ograniczenia metody

Do najciekawszych wyzwań zaproponowanego rozwiązania należą zarówno uniwersalne aspekty techniczne jak i związane z doбором dystrybucji i wersji jądra systemowego Linux. Rozwiązanie wykrywające operacje musi być dostatecznie rzetelne i w powszechnym użyciu. W innym wypadku kontrola systemu będzie zwyczajnie pełna luk, które same w sobie będą trudne do wykrycia. Efektem złego doboru rozwiązania technicznego dla funkcjonalności wykrywającej operacje na systemie plików może być utrata wsparcia deweloperskiego dla głównej wersji jądra systemu.

Największym wyzwaniem od strony technicznej jest utworzenie oprogramowania, którego logika wykrywania operacji na bieżąco, nie będzie prowadziła do wysokiego zużycia zasobów. Zbieranie dużej ilości informacji z pokaznego ciągu operacji bez wątpliwości będzie mieć wpływ na zużycie zasobów systemowych. Nie ma możliwości całkowitej eliminacji tego wpływu, można jednak podjąć kroki w doborze technologii w celu zniwelowania go.

Mimo iż postanowiłem wybrać przypadki aktywacji funkcji skanowania ścieżki, która zazębia z powszechnymi scenariuszami ataku, istnieje możliwość niewystarczającego pokrycia przypadków.

Trzeba też zaznaczyć, że dla plików skompresowanych badanie entropii ich treści może wskazywać na błędną klasyfikację w ramach metody przedstawionej w sekcji [Analiza entropii pliku](#). Nie jest to szczególnie duży problem ze względu na to, że metoda charakteryzuje się wysoką dokładnością dla plików o wielkości większej niż 32 bajty, wielkości, której przekroczenie nie jest ciężkim wyzwaniem dla plików skompresowanych.

Metoda zaprezentowana w rozdziale [Analiza podobieństwa pliku wykonywalnego](#) także nie jest jednoznaczną metodą wykrycia zagrożenia. Cytując abstrakt pracy „A Framework for Analyzing

¹https://github.com/mitre-attack/attack-datasources/blob/main/docs/images/relevant_data_components.jpg

Ransomware using Machine Learning” [37] : „Experimental results reported the performance i.e. the detection accuracy of ransomware samples which varied from 76% to 97% based on the ML technique used [...]”, można się spodziewać fałszywego stwierdzenia obecności ransomware w podobnym, a może i nawet rozleglejszym przedziale dokładności.

Analizując te braki trzeba mieć na uwadze, że projektowany program ma być jednym z wielu narzędzi dla administratora ale nie ma zwalniać go z obowiązku rzetelnego analizowania potencjalnych zagrożeń na systemie. Nie ma on też być rozwiązaniem typu „wszystko w jednym”. Jego celem jest mitygacja kosztów ataku poprzez poinformowanie administratora o możliwości ataku.

Rozdział 4

Projekt oprogramowania i użyte rozwiązania

4.1 Specyfikacja wymagań funkcjonalnych i нефункциональных

4.1.1 Wymagania funkcjonalne

Id: F1	Nazwa: Identyfikacja zaszyfrowanego pliku metodą pola między wykresami entropii
Warunek rozpoczęcia	Skan został zainicjowany na dowolny sposób
Warunki zakończenia	Sukces: wynik został zwrócony Porażka: wynik nie został zwrócony

Id: F2	Nazwa: Identyfikacja ransomware poprzez podobieństwo cos
Warunek rozpoczęcia	Skan został zainicjowany i sprawdzone zostało to czy plik jest typu ELF
Warunki zakończenia	Sukces: wynik został zwrócony Porażka: wynik nie został zwrócony

Id: F3	Nazwa: Analiza ruchu metodą ilości dokonanych operacji
Warunek rozpoczęcia	Zainicjowana została rutynowa kontrola
Warunki zakończenia	Sukces: czas wykonania operacji mieści się w przedziale czasowym zdefiniowanym w konfiguracji i przekracza ilość operacji zdefiniowaną w konfiguracji. Porażka: niemożliwe było odczytanie danych z bazy.

Id: F4	Nazwa: Analiza ruchu metodą ilości zmienionych nazw plików
Warunek rozpoczęcia	Zainicjowana została rutynowa kontrola
Warunki zakończenia	Sukces: czas wykonania operacji mieści się w przedziale czasowym zdefiniowanym w konfiguracji i przekracza ilość operacji zdefiniowaną w konfiguracji. Porażka: niemożliwe było odczytanie danych z bazy.

Id: F5	Nazwa: Analiza ruchu metodą powstania plików zawierających słowa kluczowe (ransom notes)
Warunek rozpoczęcia	Zainicjowana została rutynowa kontrola
Warunki zakończenia	Sukces: wykryty został plik i zidentyfikowany jako ransom note. Porażka: niemożliwe było odczytanie danych z bazy lub plik nie został wykryty mimo obecności danych o tym świadczących.

Id: F6	Nazwa: Analiza ruchu konfigurowalnym rutynowym skanem
Warunek rozpoczęcia	Zainicjowana została rutynowa kontrola
Warunki zakończenia	Sukces: skan został wykonany o czasie zdefiniowanym w konfiguracji. Porażka: skan się nie odbył w sprecyzowanym czasie lub interwale.

Id: F7	Nazwa: Wysyłanie powiadomień na serwer syslog
Warunek rozpoczęcia	Zakończony został skan
Warunki zakończenia	Sukces: raport o skanie został wysłany na serwer syslog. Porażka: raport nie został wysłany na serwer syslog.

4.1.2 Wymagania jakościowe

ID: J1	Nazwa: Komponent kolekcjonowania operacji powinien być wspierany przez najważniejsze dystrybucje
Rodzaj: przenośność	
Opis: Aby aplikacja mogła być użyteczna dla administratorów, element odpowiadający za zbieranie informacji o systemie plików musi być możliwy do użycia na popularnych dystrybucjach serwerowych systemu Linux.	
Sposób pomiaru: Technologia wykorzystywana do zbierania informacji musi być dostępna na Ubuntu 22.04.1 LTS Server, RHEL 8.8, RHEL 7.9, Open SUSE Leap 15.5 oraz SUSE Linux Enterprise Server 12.	
Możliwy wynik pomiaru: Funkcjonalność na danym systemie jest albo nie jest wspierana.	
Oczekiwanie wartości: Możliwe są tylko dwie wartości. Albo wszystkie dystrybucje wspierają funkcjonalność, albo nie. Gdy chociaż jedna dystrybucja nie wspiera funkcjonalności, wymóg jakościowy nie został spełniony.	

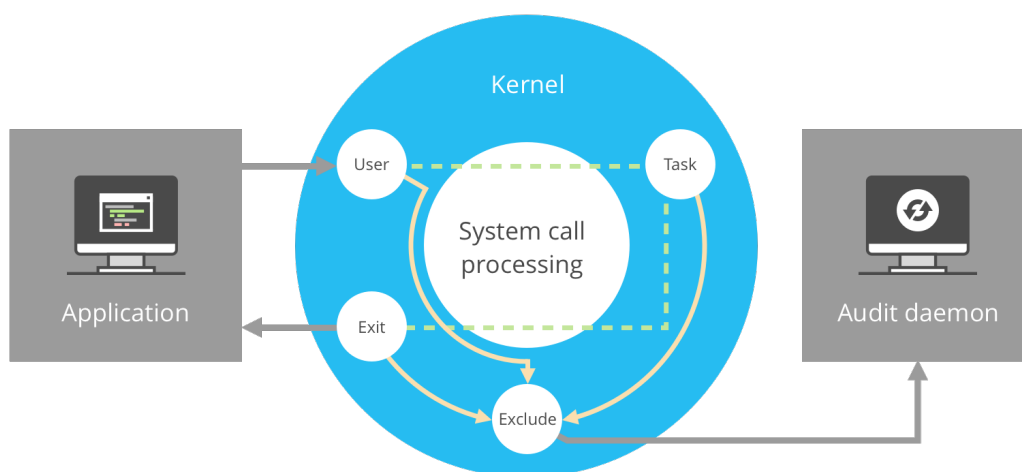
4.1. Specyfikacja wymagań funkcjonalnych i нефункциональных

ID: J2	Nazwa: Instalacja musi być bezinwazyjna
Rodzaj: przydatność funkcjonalna	
Opis: Instalacja nie może wymagać instalacji dodatkowych pakietów, zmiany integralnych ustawień systemowych ani ponownego uruchamiania systemu.	
Sposób pomiaru: Należy przejść przez proces instalacji i sprawdzić, czy będzie on wymagał ponownego uruchomienia systemu lub przeładowania modułów jądra systemu.	
Możliwy wynik pomiaru: Operacja uruchomienia ponownego i przeładowania modułów jądra systemu jest albo nie jest potrzebna przy instalacji.	
Oczekiwanie wartości: Możliwe są tylko dwie wartości. Albo proces wymaga naruszenia działania systemu albo nie. Jeśli wcześniej wymienione działania są potrzebne w procesie instalacji, wymóg jakościowy nie został spełniony.	

ID: J3	Nazwa: Niskie zużycie zasobów
Rodzaj: efektywność wydajnościowa	
Opis: Aplikacja nie może nadwyręzać zasobów systemu w sposób, który znacząco zmniejszałby jego możliwości obliczeniowe.	
Sposób pomiaru: Aplikacja powinna nie zużywać więcej niż 15% CPU dla 180 tysięcy operacji na systemie plików. Należy doprowadzić system do wykonania 180 tysięcy operacji z pomiarem zużycia CPU. Następnie powtórzyć go 10 razy i wyciągnąć medianę.	
Możliwy wynik pomiaru: Zużycie mierzymy od rozpoczęcia pierwszej do zakończenia ostatniej operacji. Liczy się najwyższe zużycie z całego przedziału czasowego.	
Oczekiwanie wartości: Mediana może przekroczyć zużycie maksymalne najwyżej o 1.5 %. W innym wypadku wymóg jakościowy nie został spełniony.	

4.2 Sposób zbierania i przetwarzania statystyk systemu plików

Mając na uwadze wymagania z sekcji Specyfikacja wymagań funkcjonalnych i niefunkcjonalnych, jako silnik obserwowania operacji na systemie plików wybrałem, przedstawiony w podsekcji drugiej podrozdziału Monitorowanie zmian na systemie plików - Linux Auditing Framework.



Rysunek 27. Diagram przedstawiający sekwencję działań wykonywanych w trakcie wykonywania operacji z równoległym audytem¹.

Na diagramie wyżej ukazany został bardzo ciekawy aspekt tej technologii. Mianowicie to, że zanim diagram stanów dla wywołania systemowego dobiegnie końca to *już*, zostanie wygenerowany raport z audytu. Jest to moim zdaniem jedna z mocniejszych stron tego rozwiązania. Pozwala on na zniwelowanie narzutu związanego z odczytaniem informacji o operacji, tym samym zmniejszając czas reakcji wykrycia ataku. Dodatkowo jest to popularne i szeroko wspierane rozwiązanie o czym świadczy chociażby obecność specjalnych instrukcji obsługi dla tej technologii na stronach RedHata² czy OpenSuse³. W dużej mierze obsługa wymaga wyczytania informacji o operacji z serii logów generowanych w ramach raportu.

```

1  type=SYSCALL msg=audit(1364481363):comm="cat" exe="/bin/cat"
2  type=CWD msg=audit(1364481363): cwd="/home/shadowman"
3  type=PATH msg=audit(1364481363): item=0 name="/etc/ssh/sshd_config"
4  type=PROCTITLE msg=audit(1364481363) : proctitle=6361740
  
```

Listing 7. Przykładowy format treści raportu z audytu. Na potrzeby estetyki prezentacji wyciąłem z niego trochę informacji.

¹<https://selectel.ru/blog/en/2017/06/08/auditing-system-events-linux/>

²https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/sec-understanding_audit_log_files

³<https://doc.opensuse.org/documentation/leap/archive/42.3/security/html/book.security/cha.audit.comp.html>

W listingu 7 przedstawiony jest format treści. Do najważniejszych informacji jakie można z nich wydobyć należą:

- czas dokonania operacji,
- typ wywołania systemowego użytego w operacji,
- ścieżka do pliku wykonywalnego w ramach którego dokonano operacji,
- identyfikator użytkownika i jego grupy,
- pliki które były argumentami wywołania operacji.

Dodatkowo można sprecyzować naturę operacji w konfiguracji auditd. Przykładowo dana jest konfiguracja:

```
1 $ sudo auditctl -a exit,always -F dir=/dir -F perm=w -F key=WRITE
2 $ sudo auditctl -a exit,always -F dir=/dir -F perm=r -F key=READ
3 $ sudo augenrules
```

Listing 8. Konfiguracja zasad audytowania.

Pierwsza linijka odpowiada za wyczytywanie operacji typu write o kluczu WRITE dla ścieżki /dir i vice versa dla drugiej linijki z operacją READ. Następnie aby zmiany weszły w życie należy użyć augenrules. Jak więc widać konfiguracja nie jest szczególnie skomplikowana co jest również dużą zaletą tego rozwiązania.

Ostatnią cechą wartą uwagi jest to, że auditd może zostać skonfigurowany w taki sposób, że informacje z raportów są wysyłane poprzez wybrane gniazdko UNIXowe o szczegółowo dostosowanych parametrach. Jednym z najbardziej wpływających na bezpieczeństwo atrybutów jest direction, który sprawia, że nie można żadnych danych do gniazdka wprowadzić, jedynie wyczytać. Opcja ta, nawiasem mówiąc, jest jednym z gwarantów wiarygodności informacji o systemie plików. W niżej przedstawionej konfiguracji gniazdko, na które będą kierowane informacje to /var/run/dispatcher. Jego atrybuty zostały ustawione w taki sposób, że użytkownik ma możliwości zapisu i odczytu gniazdka, grupa ma wyłącznie prawo odczytu, a inni użytkownicy nie mają żadnych do niego praw.

```
1 active = yes
2 direction = out
3 path = builtin_af_unix
4 type = builtin
5 args = 0640 /var/run/dispatcher
6 format = string
```

Listing 9. Konfiguracja opcji raportowania. Więcej informacji można znaleźć na stronie RedHatowej dokumentacji ⁴.

⁴https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/chap-system_auditing

4.3 Wybór technologii i narzędzi programistycznych

4.3.1 Zbieranie informacji z audytu

Aby zbieranie informacji o systemie było możliwie szybkie i wydajne pamięciowo, rozważyłem dwa języki - C oraz Rust. Dużą zaletą C był fakt istnienia bibliotek służących do efektywnego kolejkowania i parsowania informacji z audytu⁵. Niestety po bliższej inspekcji okazało się, że w implementacji kolejki następował wyciek pamięci. W związku z tym oraz prywatną chęcią bliższego zapoznania się z językiem Rust, ostatecznie wybrałem drugą opcję. Rust pozwala na stworzenie szybkiej i wydajnej aplikacji natywnej, jednocześnie gwarantując bezpieczeństwo pamięci. Głównie z tego powodu wydał mi się perfekcyjnym wyborem. Aby mieć możliwość asynchronicznej obsługi wydarzeń z raportu, skorzystałem z frameworku Tokio⁶. Pozwala on na skalowalną i dynamiczną obsługę asynchronicznych zdarzeń i tym samym efektywne przetwarzanie danych. Bezpośrednim powodem użycia rozwiązania opartego na wielowątkowości było dokonanie swego rodzaju „load balancingu” przetwarzanych informacji z auditd w celu mitygacji opóźnień w dostarczeniu danych wejściowych do skanowania dla dużej ilości operacji.

```
1 #[tokio::main]
2 async fn main() -> Result<(), Box<dyn std::error::Error>> {
3     let configs = configure(SETTINGS_ADDRESS)?;
4     simple_logger::init_with_level(match configs.log_level {
5         LogSettings::Debug => Level::Debug,
6         LogSettings::Info => Level::Info,
7     })?;
8     log::debug!("Loaded settings from: {}", SETTINGS_ADDRESS.cyan());
9     // reszta kodu ...
10 }
```

Listing 10. Korzystanie z możliwości tokio jest zaskakująco proste. Poza znajomością obsługi wątków w standardzie języka, wymaga ono jedynie użycia nagłówka nad funkcją main. Fragment kodu pochodzi z pliku main.rs w mojej pracy.

4.3.2 Logika biznesowa

Komponent odpowiadający za logikę biznesową napisany został w Javie. Był to wybór kierowany pragmatyzmem na który złożyły się: mój osobisty stopień zaawansowania w tym języku, popularność Javy, będącej gwarancją na łatwy rozwój projektu w przyszłości i jej status de facto standardu aplikacji enterprise oraz międzyplatformowość.

Postanowiłem skorzystać z frameworku Spring Boot⁷ jako naczelnego spoiwa projektu. Spring Boot oferuje wiele udogodnień przydatnych do implementacji wysokiej jakości oprogramowania. Posiada on też szeroki ekosystem tzw. starterów, czyli pakietów zależności skupiających się na różnych

⁵<https://github.com/linux-audit/audit-userspace/tree/master/auparse>

⁶<https://tokio.rs/>

⁷<https://spring.io/projects/spring-boot>

funkcjonalnościach np. bazach danych. Szczególnie zależało mi na możliwości korzystania z gotowej implementacji harmonogramów wywołań funkcji i kontenera kontekstu.

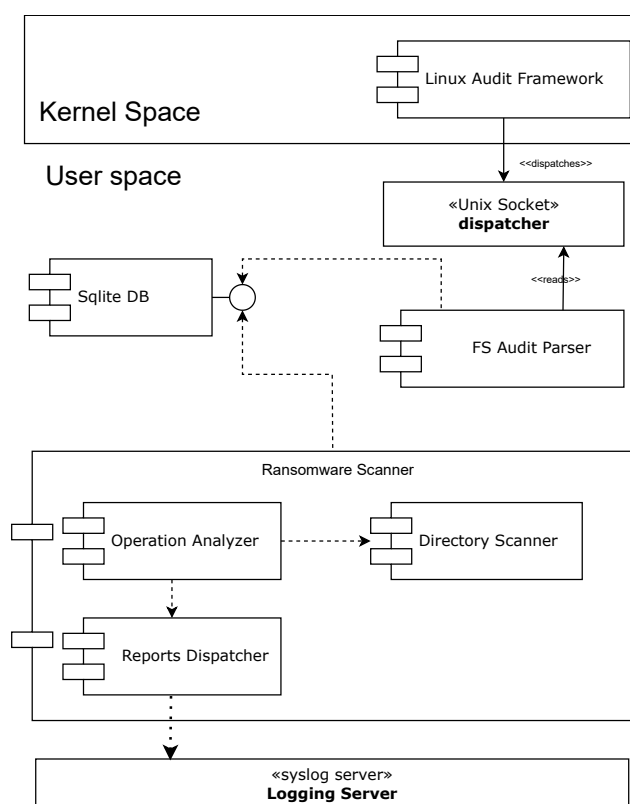
4.3.3 Baza danych

Przy wyborze bazy danych kierowałem się założeniem, że narosnąć może potrzeba dużej ilości zapisów i odczytów małej ilości danych. Jako, że nie chciałem uzależniać aplikacji od dodatkowych usług, które musiałyby być serwowane na „bierząco”, zdecydowałem się na bazę SQLite.

Technologia ta posiada wiele zalet, między innymi wymieniony wcześniej brak potrzeby serwowania danych na bazie kolejnego procesu oraz wysoką wydajność dla małej ilości zapisywanych danych. Podejmując ten wybór zainspirowałem się programem Calibre, który przechowuje informacje o ebookach w bazie SQLite-owej.

4.4 Architektura aplikacji

Najważniejsze z poziomu architektury aplikacji było w moim mniemaniu podzielenie logiki zbierania informacji o operacjach od logiki wykrywania zagrożenia. Celem tego zabiegu było pozostawienie możliwości łatwej zmiany technologii wysoce zależnych od implementacji, wersji czy dystrybucji systemu operacyjnego.

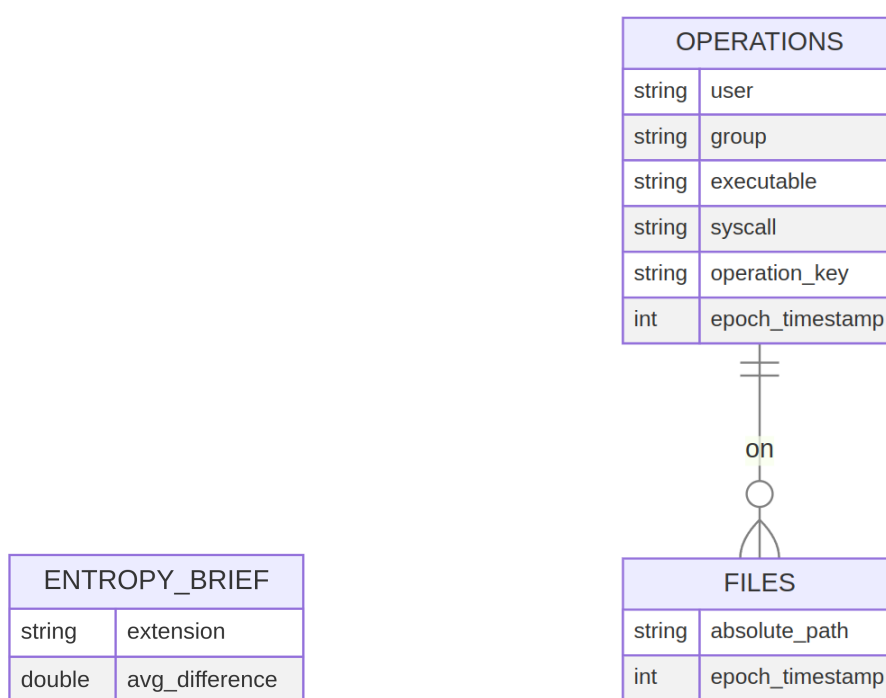


Rysunek 28. Diagram obrazujący zależności pomiędzy komponentami i źródłami informacji. Komponent FS Audit Parser nie komunikuje się bezpośrednio z komponentem Ransomware Scanner.

Komponent FS Audit Parser odpowiada za zbieranie i parsowanie informacji przesyłanych przez auditd, będący częścią Linux Audit Framework, a następnie zapisuje je w bazie SQLite.

Komponent Ransomware Scanner enkapsuluje w sobie logikę aktywowania i przeprowadzania skanu w celu wykrycia ataku. Informacje o systemie są zbierane z bazy danych. Po przeprowadzeniu skanu wysyła on raport do serwera aplikacji syslog.

FS Audit Parser dokonuje wyłącznie zapisów do bazy, a Ransomware Scanner wyłącznie odczytu. Architektura aplikacji zezwala na dowolną wymianę komponentów nie będących powiązanych bezpośrednio z jej logiką biznesową jak baza danych albo metoda zczytywania wiadomości o systemie plików.



Rysunek 29. Schemat bazy danych

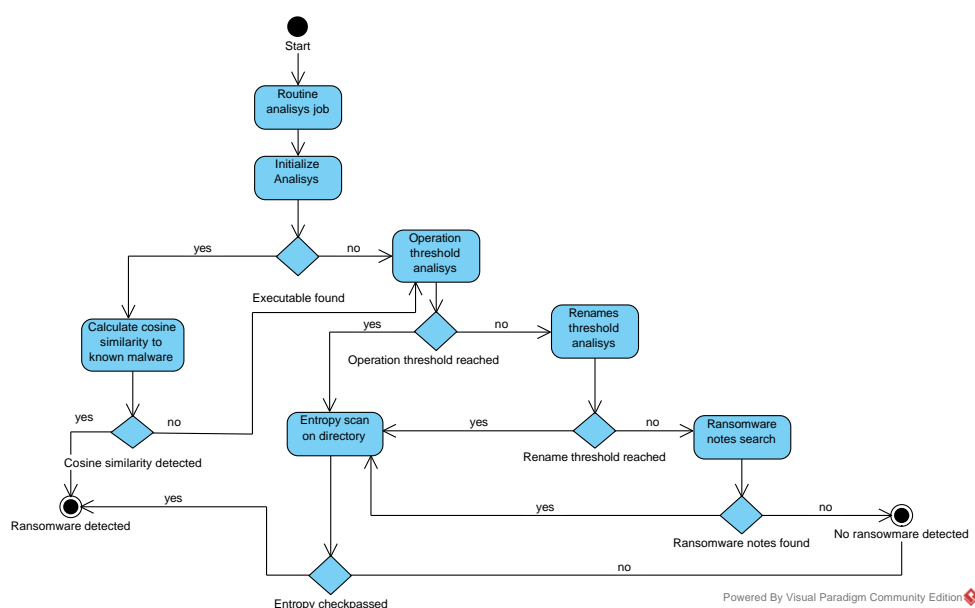
Projekt bazy danych jest bardzo prosty i oparty na podstawowych informacjach potrzebnych do implementacji logiki wykrywania ataku zdefiniowanych w sekcji *Wybór odpowiednich statystyk i metryk do analizy*. Baza danych ma agregować funkcję przekazywania informacji między komponentami i zapisywaniu ich w celu prześledzenia danych historycznych na niewielką skalę.

Projekt architektury ze względu na silnie powiązaną dziedzinę z kontekstem technologicznym, nosi znamień pokrewnej do architektury wdrożenia. Prywatnie uważam, że nie jest to problemem gdyż powszechnie stosowane rozwiązania w cyberbezpieczeństwie *nie mogą* pozwolić sobie na zupełne wyabstrachowanie swojej architektury od dziedziny technologicznej jaką mają obejmować.

4.5 Zabezpieczenia i uwierzytelnianie w systemie

Konfiguracja `auditd` na którym opiera się opisana w tej pracy aplikacja wymaga uprawnień super usera i co ważniejsze może zostać doprowadzona do stanu nienanuraszolności w bieżącym czasie uruchomienia. W tym ustawieniu wszystkie zasady obserwacji oraz watchery⁸ są niezmiennalne bez restartu systemu. Jest to bardzo duży plus, gdyż wywyższa to zmiany zasad w audycie do rangi decyzji organizacyjnej. W zależności konfiguracji kanału wyjściowego logów, komponent FS Audit Parser może wymagać uruchomienia w trybie super usera. Jest możliwość ominięcia tego problemu poprzez odpowiednie ustawienie uprawnień odczytu dla gniazdka UNIXowego, którego przykład został pokazany w listingu 9 sekcji Sposób zbierania i przetwarzania statystyk systemu plików. Sprawia to, że ponad konfigurację wstępną, administrator nie będzie miał potrzeby ingerencji w działanie systemu jako super user, co jest zdecydowanie czynnikiem zmniejszającym ryzyko powstania potencjalnych luk zabezpieczeń.

4.6 Implementacja algorytmów wykrywających podejrzane działania



Rysunek 30. Schemat analizy operacji i plików w obserwowanej ścieżce.

Powyżej przedstawiony diagram opisuje ciąg zdarzeń następujących podczas regularnie ustalonej analizy operacji na systemie plików. Brane pod uwagę jest kilka charakterystycznych scenariuszy opisanych w sekcji Charakterystyka typowych zmian w systemie plików podczas ataku ransomware. Na podstawie analizy kolejno wykonywana jest bardziej przystosowana do zaobserwowanego zjawiska metoda.

⁸Poglądowy wątek z forum w którym użytkownik chce wyłączyć watcher bez ponownego uruchamiania: <https://serverfault.com/questions/586230/disable-auditd-immutable-mode-without-rebooting>

Rozdział 5

Testowanie i walidacja

5.1 Metodologia testowania

5.1.1 Warunki testowe i zastosowane środki bezpieczeństwa

Ze względu na mój relatywny brak doświadczenia w sferze cyberbezpieczeństwa postanowiłem zachować możliwie największe środki ostrożności podczas wykonywania scenariuszy ataku. W szczególności tyczy się to infekcji prawdziwym wirusem ransomware.

Moim stanowiskiem pracy był laptop Asus TUF Gaming FX505D z 8 GB ramu i 512 GB pamięci NVME z zainstalowanym Linuksem Proxmox VE 8.1¹. Proxmox jest dystrybucją specjalizującą się w zarządzaniu wirtualną infrastrukturą (na rodzaj VmWare). Posiada ona wygodny interfejs graficzny dostępny przez sieć oraz wiele udogodnień z dziedziny wirtualizacji.

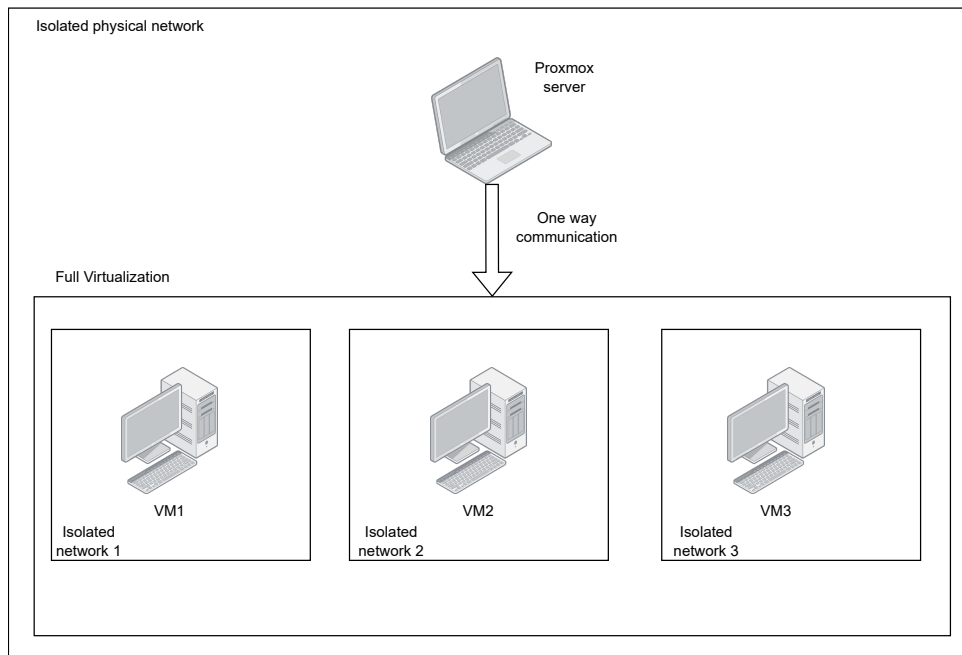
Wszystkie testowane maszyny wirtualne mają zainstalowanego Ubuntu Server 22.04 LTS². Każda z wirtualnych maszyn ma przydzielone 32 GB pamięci twardej, 8GB pamięci RAM oraz 8 procesorów. Jako hypervisor korzystałem z QEMU³ bez KVM⁴. Mimo iż użycie KVM przyspieszyłoby działanie maszyn wirtualnych, postanowiłem z niego nie korzystać z powodu braku konkretnych dowodów na absolutną odporność KVM switcha na oprogramowanie złośliwe. Dodatkowym środkiem bezpieczeństwa było zupełne wyizolowanie wirtualnych maszyn we własnych podsieciach na maszynie, która sama także była kompletnie fizycznie odizolowana od sieci zewnętrznych. Zasady audytu na maszynach zostały utworzone w trybie „immutable”, tym samym nie mogły zostać zmienione bez ponownego uruchomienia systemu.

¹Obraz systemu został pobrany ze strony <https://www.proxmox.com/en/downloads>. SHA256 obrazu wynosiło 9018a17307ad50eb9bf32a805d0917d621499363ef87b0b477332ed9f9d7dcc1.

²<https://ubuntu.com/download/server>

³<https://www.qemu.org/>

⁴https://linux-kvm.org/page/Main_Page



Rysunek 31. Architektura infrastruktury testowej.

5.1.2 Ustalenie metryki skuteczności rozwiązania

Podstawowym założeniem testów była pewność odbywania się ataku podczas działania aplikacji. W tym wypadku nie ma sensu testowanie skuteczności w formie wartości dokładności wykrywania obecności ataku, gdyż jest on stałym elementem wszystkich scenariuszy. Zamiast tego postanowiłem, że skuteczność rozwiązania będzie mierzona na podstawie *czasu wykrycia* od momentu rozpoczęcia ataku. Jest to miarodajna metryka naturalnie powiązana z potencjalnym zakresem strat w organizacji zagrożonej ransomware.

Niestety średni czas wykrycia ataku według raportu IBMu w 2018 roku wyniósł 197 dni, a w 2019 - 207 dni [38]. Aby test był możliwy do wykonania w sensownym terminie, zmuszony byłem skrócić jego czas. W związku z tym postanowiłem obrać subiektywnie wybraną miarę, umotywowaną niewielkim rozmiarem systemów użytych w testach. Mianowicie - **jeśli na 10 godzin od rozpoczęcia ataku zostanie wykryte potencjalne zagrożenie, to test został zakończony sukcesem** w przeciwnym wypadku dojdzie do porażki. W ramach testu aplikacja będzie aktywna w tle jeszcze przed rozpoczęciem ataku.

5.2 Scenariusze testowe symulujące ataki ransomware

5.2.1 Improvizowany atak z kompresowaniem

Jako podstawowy scenariusz postanowiłem dokonać ataku za pomocą archiwizacji z szyfrowaniem w folderze z milionem plików. Atak miał na celu zaszyfrowanie wyłącznie plików z rozszerzeniem `.txt`. W tym samym miejscu obecne były również pliki o innych rozszerzeniach.

```

1 $ zip --encrypt files.zip *.txt
2 $ rm -f *.txt

```

Listing 11. Komenda użyta do wykonania "ataku".

Warto dodać, że komenda nie wymagała przywilejów super usera. Istnieje więc możliwość wystąpienia podobnego ataku w organizacji w której doszło do wycieku danych kont pracowników bez przywilejów sudo.

```

1 2023-12-17T22:33:07.316 z INFO Inserting {"user":"userA","group":"
  userA","executable":"/usr/bin/rm","syscall":"unlinkat","timestamp":"
  1702852387","key":"WRITE"}
2 2023-12-17T22:33:07.351 z WARN Blocking error while reading from
  socket
3 2023-12-17T22:33:09.762 z INFO Unix stream is readable.
4 2023-12-17T22:33:09.762 z INFO Opening an Sqlite connection
5 2023-12-17T22:33:09.763 z INFO Inserting {"user":"userA","group":"
  userA","executable":"/usr/bin/bash","syscall":"openat","timestamp":"
  1702852389","key":"READ"}
6 2023-12-17T22:33:09.781 z INFO Opening an Sqlite connection
7 2023-12-17T22:33:09.792 z WARN Blocking error while reading from
  socket
8 2023-12-17T22:33:12.999 z INFO Unix stream is readable
9 2023-12-17T22:33:12.999 z INFO Opening an Sqlite connection
10 2023-12-17T22:33:12.999 z INFO Inserting {"user":"userA","group":"
  userA","executable":"/usr/bin/zip","syscall":"openat","timestamp":"
  1702852392","key":"WRITE"}
11 2023-12-17T22:33:13.028 z INFO Opening an Sqlite connection
12 2023-12-17T22:33:13.053 z INFO Unix stream is readable.
13 2023-12-17T22:33:13.053 z INFO Opening an Sqlite connection
14 2023-12-17T22:33:13.053 z INFO Inserting {"user":"userA","group":"
  userA","executable":"/usr/bin/zip","syscall":"unlink","timestamp":"
  1702852393","key":"WRITE"}
15 2023-12-17T22:33:13.100 z INFO Opening an Sqlite connection

```

Listing 12. Fragment logów z komponentu zbierającego informacje o systemie plików.

Logi z części aplikacji związanej ze zbieraniem informacji o systemie plików treściwie ukazują zakres działania tego komponentu. W trybie informacyjnym można zauważyć przewijanie się linijek z informacjami o odczytanych operacjach. Dodatkowo można zauważyć status odczytu z gniazda auditd oraz informacje o zapisie danych o operacji do bazy SQLite.

```
1      <9>1 2023-12-18T18:11:20.485Z
2      test-hostname
3      ransomware-scanner
4      - - -
5  Strategy evaluations: {
6      "Executable": "Low",
7      "Threshold": "High",
8      "Rename": "Low",
9      "RansomNote": "Low"}
10 File risk evaluation {
11     "/home/userA/box/testfile.png": "Medium",
12     "/home/userA/box/h1.csv": "Low",
13     "/home/userA/box/files.zip": "High",
14     "/home/userA/box/h2.csv": "Low"}
```

Listing 13. Raport wygenerowany ze skanera, który pozwoliłem sobie delikatnie sformatować aby widać było lepiej jego treść.

W raporcie „z autopsji” można zobaczyć, że jedyną metodą, która wykryła podejrzany ruch na maszynie była strategia z przekroczeniem ustalonej ilości operacji na jednostkę czasu. W wypadku tej maszyny było to 300 operacji na 100 ms. Czas wykrycia pokrył się z interwałem inicjalizacji analizy i wyniósł ok **167 ms**.

We fragmencie zatytułowanym „File risk evaluation” każdy plik określony swoją absolutną ścieżką ma ustaloną wartość ewaluacji. Dla plików wykonywalnych oznacza to jak wysokie jest ryzyko tego, że program jest oprogramowaniem złośliwym. Dla wszystkich innych rodzajów plików wyznaczone jest ryzyko zaszyfrowania go. Jak widać plik `files.zip` jako jedyny został oznaczony jako plik wysokiego ryzyka bycia zaszyfrowanym.

5.2.2 Atak Ransom EXX

W drugim scenariuszu dokonałem na maszynie testowej ataku wirusem Ransom EXX. Scenariusz zakłada użycie **prawdziwego** wariantu wirusa⁵. Próbkę którą wykorzystałem ma wartość funkcji SHA-256 równą `196eb5bfd52d4a538d4d0a801808298faadec1fc9aeb07c231add0161b416807`. Wirus w tej wersji to plik wykonywalny (ELF) którego ciekawymi aspektami jest brak sekcji `.note.gnu.build-id` o której była mowa w sekcji [Krótka charakterystyka plików wykonywalnych](#). Postawiona przeze mnie w tamtej sekcji propozycja częściowej identyfikacji plików wykonywalnych poprzez analizę ich nagłówka miała tutaj swoje ograniczone zastosowanie. Zanim rozpocząłem atak, umieściłem w ścieżce instalacyjnej skanera wydzieloną część `.data` wirusa z wariantu o SHA-256 równym `aa1ddf0c8312349be614ff43e80a262f`, aby program mógł skorzystać z niej przy liczeniu podobieństwa cosinusowego. Atak został wywołany z obserwowanej ścieżki, której prefixem jest `/home`.

⁵Na własną odpowiedzialność można go zdobyć ze strony https://github.com/Gi7w0rm/RansomExx_samples_and_related_artifacts


```
1      <9>1 2023-12-19T18:28:55.009Z
2      test-hostname
3      ransomware-scanner
4      - - -
5      Strategy evaluations: {
6          "Rename": "Low",
7          "Threshold": "Low",
8          "Executable": "High",
9          "RansomNote": "Low"}
10     File risk evaluation {
11         "/home/userA/box/documentFile.txt": "Low",
12         "/home/userA/box/testfile.png": "Low",
13         "/home/userA/box/h1.csv": "Low",
14         "/home/userA/box/h2.csv": "Low"
15         "/home/maciek/box/exx196": "High"}
```

Listing 14. Pierwszy raport z ataku Ransom EXX.

Zanim plik wykonywalny został aktywowany, jego obecność została oznaczona jako ryzykowna. Można więc powiedzieć, że test został zakończony sukcesem. Mimo to pozostawiłem maszynę do następnego dnia. Co ciekawe wg. raportu w czasie 16 godzin od rozpoczęcia ataku żaden z plików w /home nie został zaszyfrowany. Reguły audytowania nie zostały naruszone, podobnie żaden z komponentów aplikacji. Dodatkowo pliki, które widać w raporcie posiadają rozszerzenia formatów będących typowym celem ataków. Jest to prawdopodobnie spowodowane powolnym działaniem wirusa. Mimo to, podejście polegające na wykrywaniu „podejrzanych” plików i plików zaszyfrowanych znalazło tu swoje zastosowanie.

```
1      <9>1 2023-12-20T10:40:13.344Z
2      test-hostname
3      ransomware-scanner
4      - - -
5      Strategy evaluations: {
6          "Rename": "Low",
7          "Threshold": "Low",
8          "Executable": "High",
9          "RansomNote": "Low"}
10     File risk evaluation {
11         "/home/userA/box/documentFile.txt": "Low",
12         "/home/userA/box/testfile.png": "Low",
13         "/home/userA/box/h1.csv": "Low",
14         "/home/userA/box/h2.csv": "Low"
15         "/home/maciek/box/exx196": "High"}
```

Listing 15. Drugi raport z ataku Ransom EXX.

5.2.3 Atak Hive

Hive jest wirusem występującym zarówno w wariantcie Windowsowym jak i Linuksowym. Wersja Linuksowa została specjalnie stworzona pod atak infrastruktury wirtualnej, między innymi infekcję maszyn wirtualnych. Do ciekawych aspektów tego wirusa należy fakt napisania go w języku GO oraz to, że jego wersja Linuksowa wedle doniesień posiada wiele błędów i niedociągnięć. W trakcie testu napotkałem kilka z nich, między innymi błąd braku możliwości rozpoczęcia programu jeśli jest on wywołany wraz z prefixem ścieżki (bez znaczenia czy relatywnej czy absolutnej). Próbką którą wykorzystałem ma wartość funkcji SHA-256 równą 713b699c04f21000fca981e698e1046d4595f423bd5741d712fd7e0bc358c771⁶. Podobnie jak wirus z poprzedniej sekcji, występuje on w formacie ELF, lecz w przeciwieństwie do niego, posiada identyfikator kompilacji. Prawdopodobnie wynika to z domyślnych ustawień kompilatora GO. Również i tu wydzieliłem sekcję `.data` w celu skojarzenia podobieństwa plików wykonywalnych, ale tym razem niestety musiałem skorzystać z treści testowanego wirusa zamiast pochodnej ze względu na brak próbek. Atak został wywołany z obserwowanej ścieżki, której prefixem jest `/home`. Podobnie jak w poprzednim scenariuszu, wirus został wykryty poprzez podobieństwo plików wykonywalnych. Z tego powodu pozostawiłem maszynę wraz z działającym na nim wirusem na okres 10 godzin.

```
1      <9>1 2023-12-20T16:16:29.564Z
2      test-hostname
3      ransomware-scanner
4      - - -
5  Strategy evaluations: {
6      "Rename": "Low",
7      "Threshold": "Low",
8      "Executable": "High",
9      "RansomNote": "High"}
10 File risk evaluation {
11     "/home/userA/box/HOW-TO-DECRYPT.txt": "High",
12     "/home/userA/box/documentFile.txt": "Low",
13     "/home/userA/box/testfile.png": "Low",
14     "/home/userA/box/h1.csv": "Low",
15     "/home/userA/box/h2.csv": "Low"
16     "/home/maciek/box/hive": "High"}
```

Listing 16. Późniejszy raport z ataku Hive.

Wyniki były zaskakujące. Według raportu, żaden z plików nie został zaszyfrowany. W obserwowanym folderze pojawił się plik „HOW-TO-DECRYPT.txt” oceniony metodą wykrywania wiadomości o okupie jako niebezpieczny. W raporcie można także wyczytać, że wirus został wykryty metodą podobieństwa cosinusowego o czym wspomniałem wcześniej.

⁶Zdobyta ze strony: <https://bazaar.abuse.ch/sample/713b699c04f21000fca981e698e1046d4595f423bd5741d712fd7e0bc358c771/>

```

1   Your network has been breached and all data is encrypted.
2   To decrypt all the data you will need to purchase our decryption
   software.
3   Please contact our sales department at:
4   <url>
5   Login: <login>
6   Password: <password>
7   Follow the guidelines below to avoid losing your data:
8   - Do not shutdown or reboot your computers, unmount external
   storages.
9   - Do not try to decrypt data using third party software. It may
   cause irreversible damage.

```

Listing 17. Fragment wiadomości o okupie. Ze względów bezpieczeństwa usunąłem wszelkie linki, loginy i hasła z listingu.

Po dokładnym sprawdzeniu systemu plików na maszynie nie znalazłem żadnych śladów działania wirusa. Zasady audytu również nie zostały naruszone. Mimo dużej ilości informacji o błędach Linuksowej odmiany tego wirusa, nie byłem w stanie ustalić jednoznacznie dlaczego oprogramowanie wygenerowało dopiero po ok. 6 godzinach informacje o okupie bez wyrządzenia widocznej szkody na systemie plików. Istnieje równa szansa, że może być to zarówno błąd oprogramowania jak i zamierzony cel (choć bardzo dziwny).

5.3 Ewaluacja skuteczności wykrywania

Scenariusz\Strategia wykrycia	Rename	Threshold	Executable	Ransom Note
Improwizowany atak z kompresowaniem	Low	High	Low	Low
Ransom EXX	Low	Low	High	Low
Hive	Low	Low	High	High

Tabela 2. Ewaluacja skuteczności strategii per scenariusz.

Można zauważyć, że najmniej skuteczna w okresie 10 godzin od rozpoczęcia ataku była strategia detekcji poprzez zmianę nazwy plików. Celem tej strategii miało być wykrycie masowego nadpisywania plików ich zaszyfrowanymi odpowiednikami. Warto zadać sobie pytanie czy w takim razie faktycznie kiedykolwiek dojdzie do takiej sytuacji, nawet w wydłużonym czasie eksperymentu, w którym ta strategia się sprawdzi. Osobiście uważam, że nie, a wręcz wydłużenie czasu eksperymentu byłoby niezgodne z podstawowym założeniem tej strategii. Z eksperymentów wynika, że jest ona zbędna albo przynajmniej wymaga modyfikacji zasady działania.

Strategia poprzez wykrywanie podejrzanych plików wykonywalnych sprawdziła się znakomicie w scenariuszach ataków prawdziwych ransomware. Wynika to z faktu użycia historycznie znanych zagrożeń. Wymaga ona jednak wyizolowania próbki podobnego wirusa. W innym wypadku nie ma

szansy się sprawdzić. Najlepiej odzwierciedla to wynik eksperymentu z improwizowanym atakiem. W tym przypadku skuteczna okazała się strategia skanowania plików oraz wykrycia dużej ilości operacji.

Scenariusz	Czy doszło do zaszyfrowania ?
Improwizowany atak z kompresowaniem	✓
Ransom EXX	✗
Hive	✗

Tabela 3. Tabela ukazująca czy doszło do zaszyfrowania plików podczas testu.

Mimo iż tylko jedna lub dwie strategie wykrywały zagrożenie w danym czasie to można uznać, że testy zakończyły się sukcesem. W każdym ze scenariuszy nie nastąpiła sytuacja w której ani jedna metoda by nie wykryła zagrożenia. Można więc wyciągnąć wniosek, że metody detekcji tego typu zagrożeń nie powinny być wykorzystywane z osobna. Wskazane jest aby urozmaicać środki bezpieczeństwa i ty samym pozostawić jak najmniejsze pole do manewru do ataku.

Odczytywanie operacji za pomocą auditd sprawiło się bardzo dobrze w każdym ze scenariuszy. Analiza logów z audytu jak i stanu bazy danych po wykonaniu testów, często dostarczały więcej informacji pozwalających zrozumieć ciąg zdarzeń na testowanej maszynie niż obserwowanie jej konwencjonalnymi środkami. Metoda ta była odporna na działania rzeczywistych ransomware w każdym ze scenariuszy.

Rozdział 6

Podsumowanie

6.1 Główne osiągnięcia pracy

Głównym celem pracy było utworzenie „proof of concept” oprogramowania analizującego ruch na systemie plików w celu wykrycia ataku ransomware. Ten cel został w dużej mierze osiągnięty. Rozwiązanie spełniło wszystkie wymagania jakościowe i funkcjonalne zdefiniowane w sekcji [Specyfikacja wymagań funkcjonalnych i niefunkcjonalnych](#). Udało się przede wszystkim utworzyć oprogramowanie będące silnikiem napędzającym dalsze metody analizy pod kątem ataku oprogramowaniem złośliwym. Wykorzystane zostały znane w branży administracji systemami metody, które bez wątpienia każdy administrator będzie w stanie w efektywny i wygodny sposób dostosować do własnych potrzeb. Przeprowadzenie praktycznych testów także było pomocne w dopracowaniu zakresu rzeczywistych wymagań jakie powinny spełniać analizowane środowiska oraz w jakim toku tego typu eksperymenty powinny być przeprowadzane. Wyniki testów wskazują na to, że założenie o ataku objawiającym się w mniej niż dobę było błędne. Kluczem w analizie zachowań wirusów ransomware jest użycie rozwiązania w warunkach praktycznych w okresie conajmniej kilku tygodni. Mimo tego niedopatrzenia udało się wyciągnąć inne interesujące wnioski na temat skuteczności i potrzeby dywersyfikacji wykorzystywanych metod.

6.2 Ograniczenia proponowanej metody

Największym ograniczeniem zaproponowanej metody jest zależność od zasad audytowania systemu. Możliwe byłoby utworzenie zupełnie odrębnego rozwiązania ale musiałoby to się wiązać z potrzebą dystrybuowania specjalnej wersji jądra systemowego lub akceptacji zmian na głównym repozytorium projektu Linux. Dodatkowo Linux Auditing Framework również sam z siebie wpływa na wydajność systemu. W związku z tym na administratorze niestety spoczywa obowiązek dostosowania zasad audytu pod względem zużycia zasobów, również biorąc pod uwagę dodatkowe rozwiązania, które się na nich opierają.

Ograniczającą także jest zależność rozwiązania operującego się na wyliczaniu różnicy w polu entropii na danych w zależności od rozpatrywanego formatu. W mojej implementacji opierałem się na danych zebranych z pracy „Differential area analysis for ransomware attack detection within mixed file datasets” [33] jednak ilość próbek, na podstawie której wyznaczono graniczne wartości różnicy pól niewielkie. Należałoby zrobić serię badań dla większej ilości formatów z większą ilością próbek. To samo tyczy się metody podobieństwa cosinusowego z dodatkową wadą wymogu analizy tekstowej kodu asemblerowego plików posiadających potencjalnie tysiące linii.

6.3 Propozycje dalszego rozwoju i doskonalenia systemu

Aby można było rozwinąć koncept przedstawionego oprogramowania w godny uwagi produkt, należałoby zaimplementować możliwość utworzenia harmonogramu zadań. Zadaniami w tym modelu byłyby skany poszczególnymi metodami lub dowolnej ich kombinacją. Z punktu widzenia administracji systemami, dobrym dodatkiem byłby także interfejs graficzny i klient CLI ułatwiający automatyzację. Należałoby także, tak jak wymieniłem w poprzedniej sekcji, dokonać badań na większej ilości zaszyfrowanych plików o szerokiej gamie rozszerzeń, a także sformułować i zaimplementować inne rodzaje algorytmów analizujących ruch na systemie plików.

Bibliografia

- [1] THE RADICATI GROUP, I., *Email Statistics Report 2019 2023 Executive Summary*, English, lut. 2019. adr.: <https://radicati.com/wp/wp-content/uploads/2018/12/Email-Statistics-Report-2019-2023-Executive-Summary.pdf> (term. wiz. 26. 11. 2023).
- [2] *Data Never Sleeps 10.0 | Domo*, en. adr.: <https://www.domo.com/data-never-sleeps> (term. wiz. 26. 11. 2023).
- [3] Petrosyan, A., „Worldwide number of ransomware attacks 2022,” en, spraw. tech. adr.: <https://www.statista.com/statistics/1315826/ransomware-attacks-worldwide/> (term. wiz. 26. 11. 2023).
- [4] Petrosyan, A., „Number of ransomware attempts per year 2022,” en, spraw. tech. adr.: <https://www.statista.com/statistics/494947/ransomware-attempts-per-year-worldwide/> (term. wiz. 26. 11. 2023).
- [5] Herjavec, G., „Healthcare Cybersecurity Report Q4 2021,” spraw. tech. adr.: <https://www.herjavecgroup.com/wp-content/uploads/2021/10/2021-Healthcare-Cybersecurity-Report.pdf> (term. wiz. 26. 11. 2023).
- [6] Check Point Research, T., *Check Point Research: Third quarter of 2022 reveals increase in cyberattacks and unexpected developments in global trends*, en-US, paź. 2022. adr.: <https://blog.checkpoint.com/2022/10/26/third-quarter-of-2022-reveals-increase-in-cyberattacks/> (term. wiz. 26. 11. 2023).
- [7] Petrosyan, A., *Global average cost of a data breach 2023*, en. adr.: <https://www.statista.com/statistics/987474/global-average-cost-data-breach/> (term. wiz. 26. 11. 2023).
- [8] *Stop Ransomware | CISA*, en. adr.: <https://www.cisa.gov/stopransomware> (term. wiz. 26. 11. 2023).
- [9] Young, A. i Yung, M., „Cryptovirology: extortion-based security threats and countermeasures,” w *Proceedings 1996 IEEE Symposium on Security and Privacy*, 1996, s. 129–140. DOI: 10.1109/SECPRI.1996.502676.
- [10] Czarnecki, M., „Oto Marcus Hutchins, 22-letni Brytyjczyk, który zatrzymał światowy cyberatak,” pl, *wyborcza.pl*, maj 2017. adr.: <https://wyborcza.pl/7,75399,21816943,oto-marcus-hutchins-22-letni-brytyjczyk-ktory-zatrzymal-swiatowy.html> (term. wiz. 26. 11. 2023).

- [11] NHS, *RedBoot - Ransomware that Encrypts the Hard Drive Permanently*, en. adr.: <https://digital.nhs.uk/cyber-alerts/2017/cc-1673> (term. wiz. 29.11.2023).
- [12] Young, A. L. i Yung, M., „Cryptovirology: The Birth, Neglect, and Explosion of Ransomware,” *Commun. ACM*, t. 60, nr. 7, s. 24–26, czer. 2017, ISSN: 0001-0782. DOI: 10.1145/3097347. adr.: <https://doi.org/10.1145/3097347>.
- [13] „Virus Bulletin, January 1990,” en, 1990.
- [14] Tromer, E., „Cryptanalysis of the Gpcode.ak ransomware virus,” en,
- [15] *Arhiveus Ransomware Trojan Threat Analysis*, en. adr.: <https://www.secureworks.com/research/arhiveus> (term. wiz. 05.12.2023).
- [16] *CryptoLocker Ransomware Information Guide and FAQ*, en-us. adr.: <https://www.bleepingcomputer.com/virus-removal/cryptolocker-ransomware-information> (term. wiz. 06.12.2023).
- [17] *Office of Public Affairs | U.S. Leads Multi-National Action Against “Gameover Zeus” Botnet and “Cryptolocker” Ransomware, Charges Botnet Administrator | United States Department of Justice*, en, czer. 2014. adr.: <https://www.justice.gov/opa/pr/us-leads-multi-national-action-against-gameover-zeus-botnet-and-cryptolocker-ransomware> (term. wiz. 06.12.2023).
- [18] *‘Operation Tovar’ Targets ‘Gameover’ Zeus Botnet, CryptoLocker Scourge – Krebs on Security*, en-US, czer. 2014. adr.: <https://krebsonsecurity.com/2014/06/operation-tovar-targets-gameover-zeus-botnet-cryptolocker-scurge/> (term. wiz. 06.12.2023).
- [19] Hern, A., „Ransomware hackers steal plans for upcoming Apple products,” en-GB, *The Guardian*, kw. 2021, ISSN: 0261-3077. adr.: <https://www.theguardian.com/technology/2021/apr/22/ransomware-hackers-steal-plans-upcoming-apple-products> (term. wiz. 06.12.2023).
- [20] McMillan, R., „Ransomware Attack Affecting Likely Thousands of Targets Drags On,” en-US, *Wall Street Journal*, lip. 2021, ISSN: 0099-9660. adr.: <https://www.wsj.com/articles/ransomware-group-behind-meat-supply-attack-threatens-hundreds-of-new-targets-11625285071> (term. wiz. 06.12.2023).
- [21] *Kaseya was fixing zero-day just as REvil ransomware sprung their attack*, en-us. adr.: <https://www.bleepingcomputer.com/news/security/kaseya-was-fixing-zero-day-just-as-revil-ransomware-sprung-their-attack/> (term. wiz. 06.12.2023).
- [22] huntresslabs, *Critical Ransomware Incident in Progress*, Reddit Post, lip. 2021. adr.: www.reddit.com/r/msp/comments/ocggbv/critical_ransomware_incident_in_progress/ (term. wiz. 06.12.2023).
- [23] *New attack vectors for the DarkSide ransomware gang*, en. adr.: <https://www.acronis.com/en-sg/cyber-protection-center/posts/new-attack-vectors-for-the-darkside-ransomware-gang/> (term. wiz. 06.12.2023).

-
- [24] Vehabovic, A., Ghani, N., Bou-Harb, E., Crichigno, J. i Yayimli, A., „Ransomware Detection and Classification Strategies,” en, w *2022 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, Sofia, Bulgaria: IEEE, czer. 2022, s. 316–324, ISBN: 978-1-66549-749-7. DOI: 10.1109/BlackSeaCom54372.2022.9858296. adr.: <https://ieeexplore.ieee.org/document/9858296/> (term. wiz. 07.12.2023).
- [25] *Community Night SANS Secure Australia 2023 - Detecting & Hunting Ransomware Operator Tools: It Is Easier Than You Think!* | SANS Institute. adr.: <https://www.sans.org/webcasts/community-night-sans-secure-australia-2023-detecting-hunting-ransomware-operator-tools-its-easier-than-you-think/> (term. wiz. 08.12.2023).
- [26] *Ext4 Disk Layout - Ext4*. adr.: https://ext4.wiki.kernel.org/index.php/Ext4_Disk_Layout (term. wiz. 08.12.2023).
- [27] Bovet, D. P. i Cesati, M., *Understanding Linux Kernel 3rd Edition*. adr.: <https://doc.lagout.org/operating%20system%20/linux/Understanding%20Linux%20Kernel.pdf> (term. wiz. 09.12.2023).
- [28] Love, R., *Linux system programming*, en, Second edition. Beijing: O'Reilly, 2013, OCLC: ocn827267973, ISBN: 978-1-4493-3953-1.
- [29] Biancalana, A., *inotify-tools wiki*, en. adr.: <https://github.com/inotify-tools/inotify-tools/wiki/Home> (term. wiz. 10.12.2023).
- [30] *fanotify(7) - Linux manual page*. adr.: <https://man7.org/linux/man-pages/man7/fanotify.7.html> (term. wiz. 10.12.2023).
- [31] Foundation, L., *Tool Interface Standard Portable Formats Specification*. adr.: <https://refspecs.linuxfoundation.org/elf/TIS1.1.pdf> (term. wiz. 08.12.2023).
- [32] *elf(5) - Linux manual page*. adr.: <https://man7.org/linux/man-pages/man5/elf.5.html> (term. wiz. 09.12.2023).
- [33] Davies, S. R., Macfarlane, R. i Buchanan, W. J., „Differential area analysis for ransomware attack detection within mixed file datasets,” en, *Computers & Security*, t. 108, s. 102377, wrz. 2021, ISSN: 01674048. DOI: 10.1016/j.cose.2021.102377. adr.: <https://linkinghub.elsevier.com/retrieve/pii/S0167404821002017> (term. wiz. 10.12.2023).
- [34] Shannon, C. E., „A mathematical theory of communication,” *The Bell System Technical Journal*, t. 27, nr. 3, s. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.
- [35] Chen, Q. i Bridges, R. A., „Automated Behavioral Analysis of Malware: A Case Study of WannaCry Ransomware,” w *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017, s. 454–460. DOI: 10.1109/ICMLA.2017.0-119.
- [36] Salton, G. i Buckley, C., „Term-weighting approaches in automatic text retrieval,” en, *Information Processing & Management*, t. 24, nr. 5, s. 513–523, sty. 1988, ISSN: 03064573. DOI: 10.1016/0306-4573(88)90021-0. adr.: <https://linkinghub.elsevier.com/retrieve/pii/0306457388900210> (term. wiz. 10.12.2023).

- [37] Poudyal, S., Subedi, K. P. i Dasgupta, D., „A Framework for Analyzing Ransomware using Machine Learning,” w *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2018, s. 1692–1699. DOI: [10.1109/SSCI.2018.8628743](https://doi.org/10.1109/SSCI.2018.8628743).
- [38] Security, I. i Institute, P., „2019 Cost of a Data Breach Report,” en,

Spis rysunków

1	Sektory infrastruktury krytycznej, do których odnosiły się skargi IC3 ¹	10
2	Najpopularniejsze warianty wirusów ransomware, zarejestrowane w trakcie incydentów mających na celu atak infrastruktury krytycznej. Należy zauważyć, że wirus „LockBit” sprawiał najwięcej problemów. Jego wersja na system Linux nosi nazwę „LockBit Linux-ESXi Locker ” ²	11
3	Średni koszt naruszenia danych 2016-2022 ³	11
4	Globalnie zgłoszone incydenty ataków ransomware per kwartał w roku 2022 zarejestrowanych przez Check Point Research. Organizacja spekuluje, że wzrost ataków mógł być spowodowany lukami bezpieczeństwa „log4j” oraz cyberataków związanych z wojną w Ukrainie ⁴	12
5	Incydenty ransomware per sektor gospodarki ⁵	13
6	Ekran wyświetlający się po zainfekowaniu komputera przez WannaCry. Atakujący wymaga od ofiary zapłaty Bitcoinem ⁶	14
7	Ekran rozruchu przy infekcji wirusem RedBoot ⁷	14
8	Diagram sekwencji ataku ransomware.	15
9	Wiadomość ukazująca się po aktywacji wirusa „AIDS trojan” ⁸	18
10	Ekran wyświetlający się po zainfekowaniu komputera przez CryptoLocker ⁹	19
11	Miejsce w pliku binarnym agent.exe, w którym wywoływany jest MsMpeng oraz ładowany plik .dll ¹⁰	21
12	W wyniku działania wirusa tapeta użytkownika zostaje zmieniona na taką, jak widać na obrazku ¹¹	22
13	Tradycyjne antywirusy tak jak pokazany na obrazku Acronis, korzystają z metody wykrywania poprzez sygnaturę ¹²	23
14	Rola wirtualnego systemu plików w operacji kopiowania ¹³	25
15	Interakcja pomiędzy procesami a obiektami wirtualnego systemu plików ¹⁴	25
16	Bardzo uproszczony diagram komponentów LAF ¹⁵	27
17	Podział wewnętrzny pliku ELF ¹⁶	28
18	Wykres entropii od długości nagłówka dla pliku zawierającego zupełnie losowe dane.	30

19	Zestawienie wykresów entropii od długości nagłówka. Jako plik przykładowy wybrałem skrypt z sekcji Monitorowanie zmian na systemie plików.	30
20	Pole między wykresem losowego i rzeczywistego pliku o takich samych długościach. .	31
21	Skuteczność dla wybranych kryteriów klasyfikacji na długość nagłówka w bajtach ¹⁷ . .	31
22	Tabela plików tymczasowych wykorzystywanych przez wirus WannaCry ¹⁸	32
23	Tabela wyników z eksperymentu. Eksperyment pierwszy i drugi są tutaj nazwane „case 1” i „case 2”. Rankingi są wyliczone na podstawie wartości wagi TF-IDF ze wszystkich dokumentów ¹⁹	33
24	Prawdopodobieństwa wystąpienia bajtów o konkretnej wartości w pliku z listingu 3. .	35
25	Prawdopodobieństwa wystąpienia bajtów o konkretnej wartości w pliku z listingu 5. .	36
26	Najbardziej wartościowe źródła danych o ataku wg. MITRE ²⁰	39
27	Diagram przedstawiający sekwencję działań wykonywanych w trakcie wykonywania operacji z równoległym audytem ²¹	44
28	Diagram obrazujący zależności pomiędzy komponentami i źródłami informacji. Komponent FS Audit Parser nie komunikuje się bezpośrednio z komponentem Ransomware Scanner.	47
29	Schemat bazy danych	48
30	Schemat analizy operacji i plików w obserwowanej ścieżce.	49
31	Architektura infrastruktury testowej.	52

Spis tabel

1	Tabela popularnych narzędzi wykorzystywanych w atakach ransomware ²²	24
2	Ewaluacja skuteczności strategii per scenariusz.	57
3	Tabela ukazująca czy doszło do zaszyfrowania plików podczas testu.	58

Spis załączników