WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ CYBERNETYKI

Diagnostyka i tolerowanie uszkodzeń

Sprawozdanie projektu – Porównanie wydajności funkcji skrótu

inż. Bartłomiej Pawelec

inż. Maciej Talecki

inż. Hubert Makowski

Grupa: WCY24KX1S4

Prowadzący: mgr inż. Melaniuk Michał

Spis treści

1. Wstęp

1.1. Cel projektu

Celem niniejszego projektu było przeprowadzenie kompleksowej analizy porównawczej czterech wybranych funkcji skrótu: MD5, SHA-1, SHA3-512 oraz Whirlpool-512. Funkcje te należą do algorytmów kryptograficznych różnych generacji i są wykorzystywane w szerokim zakresie zastosowań, od weryfikacji integralności danych po zaawansowane systemy bezpieczeństwa informatycznego.

Z uwagi na znaczenie wydajności operacyjnej funkcji skrótu w systemach komputerowych, szczególnie w kontekście dużych wolumenów danych oraz zastosowań czasu rzeczywistego, projekt skoncentrował się na ocenie następujących parametrów:

- Latencja (czas pojedynczej operacji) – mierzono czas niezbędny do wygenerowania skrótu dla pojedynczej jednostki danych, co odzwierciedla efektywność działania funkcji w mikrooperacjach.
- Przepustowość (MB/s) – analizowano szybkość przetwarzania danych w dłuższych strumieniach, wyrażoną jako liczba megabajtów przetworzonych w jednostce czasu.
- Zużycie zasobów systemowych – rejestrowano obciążenie procesora (CPU) oraz zapotrzebowanie na pamięć operacyjną (RAM), co ma istotne znaczenie przy wdrażaniu funkcji skrótu w środowiskach o ograniczonych zasobach.
- Stabilność działania – oceniano spójność wyników czasowych dla wielu iteracji tego samego algorytmu, z wykorzystaniem wskaźników statystycznych takich jak odchylenie standardowe oraz percentyle.

1.2. Zakres badań

Badania eksperymentalne zostały przeprowadzone na dwóch typach danych:

- plikach rzeczywistych, reprezentujących typowe przypadki użycia, takie jak obrazy systemów operacyjnych w formacie ISO, dokumenty PDF, archiwa ZIP czy pliki multimedialne,
- syntetycznych buforach losowych, generowanych automatycznie o różnych rozmiarach, w celu zapewnienia jednorodnych warunków testowych i powtarzalności pomiarów.

Zastosowanie obu typów danych umożliwiło uzyskanie szerszego obrazu zachowania algorytmów – zarówno w kontekście rzeczywistego środowiska użytkownika, jak i kontrolowanych testów laboratoryjnych. Dla każdego typu danych testowano różne wielkości wejściowe (od kilku kilobajtów do kilkuset megabajtów), aby sprawdzić, jak poszczególne funkcje skrótu skalują się w zależności od rozmiaru przetwarzanych danych.

Wybór funkcji skrótu nie był przypadkowy – każda z nich reprezentuje inny poziom zaawansowania technologicznego i różne podejście projektowe:

- MD5 – klasyczny algorytm skrótu, obecnie uznawany za niebezpieczny kryptograficznie, ale wciąż stosowany w mniej krytycznych zastosowaniach.

- SHA-1 – nieco nowszy niż MD5, szeroko stosowany w przeszłości, lecz również obecnie uważany za niezalecany ze względu na znane słabości.
- SHA3-512 – jedna z najnowszych funkcji skrótu, oparta na konstrukcji Keccak, oferująca wysoki poziom bezpieczeństwa oraz unikalną strukturę wewnętrzną.
- Whirlpool-512 – mniej popularny, lecz ceniony za solidną konstrukcję kryptograficzną i odporność na znane typy ataków.

2. Charakterystyka badanych funkcji skrótu

Funkcja skrótu to deterministyczny algorytm kryptograficzny, którego zadaniem jest przekształcenie danych wejściowych o dowolnej długości w skrót o ustalonej, z góry określonej długości.

Każda bezpieczna funkcja skrótu powinna spełniać trzy podstawowe własności:

- **Jednokierunkowość** – na podstawie wygenerowanego skrótu niemożliwe powinno być odtworzenie danych wejściowych.
- **Odporność na kolizje** – niezwykle trudne powinno być znalezienie dwóch różnych danych wejściowych, które generują taki sam skrót.
- **Efekt lawinowy** – minimalna zmiana danych wejściowych (nawet pojedynczy bit) powinna skutkować znacząco różniącym się skrótem wyjściowym.

2.1. MD5

MD5 (Message Digest 5) to jedna z najstarszych powszechnie stosowanych funkcji skrótu, opracowana przez Rona Rivesta w 1991 roku.

- Długość skrótu: 128 bitów
- Zastosowania: historycznie szeroko wykorzystywana do generowania sum kontrolnych plików, weryfikacji danych, a także w starszych protokołach uwierzytelniania.
- Uwagi: Pomimo dużej szybkości działania, MD5 obecnie uznawana jest za funkcję niebezpieczną kryptograficznie z powodu znanych ataków kolizyjnych. Ze względu na niską odporność na kolizje, jej stosowanie nie jest zalecane w systemach bezpieczeństwa.

2.2. SHA-1

SHA-1 (Secure Hash Algorytm 1) została zaprojektowana przez NSA i opublikowana przez NIST w 1995 roku jako następca MD5.

- Długość skrótu: 160 bitów
- Zastosowania: wykorzystywana była m.in. w systemach kontroli wersji (takich jak Git), starszych wersjach protokołów TLS oraz w cyfrowych podpisach.
- Uwagi: Choć SHA-1 oferuje lepsze właściwości kryptograficzne niż MD5, również została uznana za podatną na ataki kolizyjne. Współczesne standardy bezpieczeństwa odchodzą od jej używania, zastępując ją silniejszymi funkcjami, takimi jak SHA-2 lub SHA-3.

2.3. SHA3-512

SHA3-512 to funkcja skrótu będąca częścią rodziny SHA-3, wybrana przez NIST w 2012 roku w ramach otwartego konkursu. Oparta jest na innowacyjnej konstrukcji Keccak, znacząco różniącej się od poprzednich rodzin SHA.

- Długość skrótu: 512 bitów
- Zastosowania: nowoczesne systemy kryptograficzne, w tym protokoły TLS/SSL, blockchain, cyfrowe podpisy i inne zastosowania wymagające wysokiego poziomu bezpieczeństwa.
- Uwagi: SHA3-512 charakteryzuje się bardzo wysoką odpornością na kolizje i inne typy ataków kryptograficznych. Mimo że jest wolniejsza od SHA-2 w niektórych implementacjach, oferuje niezależność strukturalną oraz większą odporność teoretyczną, co czyni ją perspektywicznym wyborem dla nowych systemów.

2.4. Whirlpool-512

Whirlpool to mniej znana, lecz solidna funkcja skrótu opracowana przez Vincenta Rijmena (współtwórcę AES) oraz Paulo Barreto. Bazuje na strukturze podobnej do AES, co ułatwia jej implementację sprzętową.

- Długość skrótu: 512 bitów
- Zastosowania: znajduje zastosowanie w systemach archiwizacji, zabezpieczeniach danych oraz wszędzie tam, gdzie pożądany jest długi, stabilny skrót.
- Uwagi: Whirlpool oferuje wysoką odporność na kolizje i inne typy ataków, jednak ze względu na mniejszą popularność, nie jest tak szeroko wspierana w bibliotekach kryptograficznych. Może być nieco wolniejsza od konkurencji w ogólnych implementacjach programowych, ale jej konstrukcja zapewnia duży potencjał bezpieczeństwa.

3. Metodyka badania

3.1. Środowisko badawcze i przygotowanie danych

W celu przeprowadzenia pomiarów wydajności funkcji skrótu przygotowano specjalistyczne środowisko testowe oparte na języku C# oraz platformie .NET 8.0. Do implementacji wykorzystano biblioteki:

- System.Security.Cryptography – dla funkcji MD5 oraz SHA-1,
- Org.BouncyCastle.Crypto.Digests – dla SHA3-512 oraz Whirlpool-512.

Testy przeprowadzono w systemie Windows 10 przy użyciu środowiska programistycznego JetBrains Rider. Dane testowe umieszczono w folderze TestowanePliki, w którym znajdowały się pliki o czterech różnych rozmiarach:
3 KB, 3 MB, 150 MB oraz 1,2 GB.

Do badań wykorzystano dwa typy danych:

- Pliki rzeczywiste: dokumenty PDF, obrazy ISO systemów, archiwa kontenerowe Docker, pliki binarne.
- Pliki syntetyczne: bufory bajtowe wypełnione losowymi wartościami lub wzorcem 0x00, o identycznych rozmiarach jak pliki rzeczywiste.

Wczytywanie plików do pamięci realizowano za pomocą metody File.ReadAllBytes, co pozwalało na szybkie przetwarzanie danych o wielkości poniżej 2 GB. Dane wyjściowe były porządkowane w katalogach, a dla każdego pliku testowego tworzono oddzielny folder wynikowy, w którym zapisywano zarówno dane surowe, jak i zestawienia podsumowujące.

3.2. Architektura testów i struktura wyników

Każda funkcja skrótu została zaimplementowana jako osobna metoda (MD5Hash, SHA1Hash, SHA3Hash, WhirlpoolHash), która przetwarzała dane w formacie byte[] lub Stream. Testy wydajnościowe przeprowadzono zgodnie z jednolitą architekturą, w której dla każdej kombinacji algorytmu i pliku wykonano 100 powtórzeń (iteracji).

Podczas każdej iteracji rejestrowano:

- latencję (czas wykonania pojedynczego skrótu) – z użyciem klasy Stopwatch,
- zużycie CPU – jako różnica Process.TotalProcessorTime przed i po operacji,
- zużycie pamięci RAM – na podstawie GC.GetTotalMemory(false),
- przepustowość (MB/s) – liczona jako stosunek rozmiaru danych do czasu ich przetworzenia.

Dodatkowo gromadzono dane zagregowane dla całego zestawu 100 iteracji: łączny czas procesora i przyrost zużycia pamięci.

Wyniki były zapisywane w dwóch formatach:

1. Pliki surowe (raw data) – osobne pliki CSV dla każdej funkcji (md5.csv, sha1.csv itd.), zawierające:
   Iteration, LatencyMs, CpuTimeSec, MemoryUsageMB, Przepustowość (MB/s).
2. Podsumowanie zbiorcze – plik hash_benchmark_results.csv, w którym zestawiono dane zbiorcze według pól:
   Algorytm, Typ Danych, Rozmiar Danych (MB), Średni czas wykonania (ms), Odchylenie standardowe czasu (ms), Opóźnienie P50 (ms), Opóźnienie P95 (ms), Opóźnienie P99 (ms), Przepustowość (MB/s), Przepustowość (GB/s), Zużycie CPU (s), Zużycie Pamięci (MB).

Zastosowana architektura umożliwiła analizę funkcji skrótu pod względem ich efektywności operacyjnej i stabilności działania w różnych warunkach obciążenia.

3.3. Metryki

W celu kompleksowej oceny wydajności i stabilności działania funkcji skrótu zastosowano zestaw zróżnicowanych metryk pomiarowych. Każda z nich odpowiada innemu aspektowi efektywności algorytmów i pozwala na obiektywne porównanie ich zachowania w różnych warunkach obciążenia.

- Latencja (Latency)
  Reprezentuje średni czas wykonania pojedynczej operacji skrótu dla danego pliku. Wyrażona w milisekundach (ms), stanowi podstawowy wskaźnik szybkości działania funkcji. Niższa wartość oznacza szybsze przetwarzanie pojedynczych porcji danych. Porównanie latencji pomiędzy algorytmami umożliwia wskazanie najbardziej responsywnych rozwiązań.
- Odchylenie standardowe i percentyle (StdDev, P50, P95, P99)
  Te metryki statystyczne służą do oceny stabilności działania algorytmu.
  - Odchylenie standardowe (StdDev) pokazuje, jak bardzo czasy poszczególnych iteracji odbiegają od średniej.
  - Percentyl 50 (P50) to mediana – połowa pomiarów jest poniżej tej wartości.
  - Percentyl 95 (P95) i percentyl 99 (P99) wskazują, ile wynosi czas wykonania operacji w przypadku najbardziej czasochłonnych 5% i 1% iteracji.
    Dzięki tym miarom możliwa jest analiza spójności działania i identyfikacja potencjalnych opóźnień.
- Przepustowość (Throughput)
  Mierzona w megabajtach lub gigabajtach na sekundę (MB/s, GB/s), określa efektywność funkcji w kontekście przetwarzania dużych porcji danych. Im wyższa przepustowość, tym lepiej dana funkcja radzi sobie z przetwarzaniem dużych plików w krótkim czasie. Parametr ten jest szczególnie istotny w systemach archiwizacji, backupu i analizy strumieniowej.
- Zużycie zasobów systemowych (CPU Δ, RAM Δ)
  Pomiar zużycia procesora (CPU) oraz pamięci operacyjnej (RAM) dostarcza informacji o wpływie funkcji skrótu na środowisko systemowe.

- CPU Δ (różnica w czasie procesora, w sekundach) wskazuje, ile czasu obliczeniowego zajmuje wykonanie serii operacji.
- RAM Δ (różnica zużycia pamięci, w MB) pokazuje, o ile zwiększyło się użycie pamięci w trakcie testów.
  Wskaźniki te pozwalają ocenić, jak bardzo dana funkcja obciąża zasoby i czy nadaje się do środowisk o ograniczonej wydajności.

- Skalowalność
  Mierzona na podstawie wzrostu czasu wykonania funkcji w zależności od wielkości danych wejściowych. Algorytm uznaje się za dobrze skalowalny, jeśli czas przetwarzania rośnie liniowo względem rozmiaru pliku. Skalowalność jest kluczowa w kontekście przetwarzania dużych wolumenów danych, np. w systemach Big Data czy bazach danych.

Zestawienie tych metryk umożliwia wieloaspektową analizę funkcji skrótu – nie tylko pod kątem ich szybkości, ale także stabilności, efektywności przy dużych zbiorach danych oraz wpływu na zasoby systemowe.

3.4. Kluczowe fragmenty implementacji

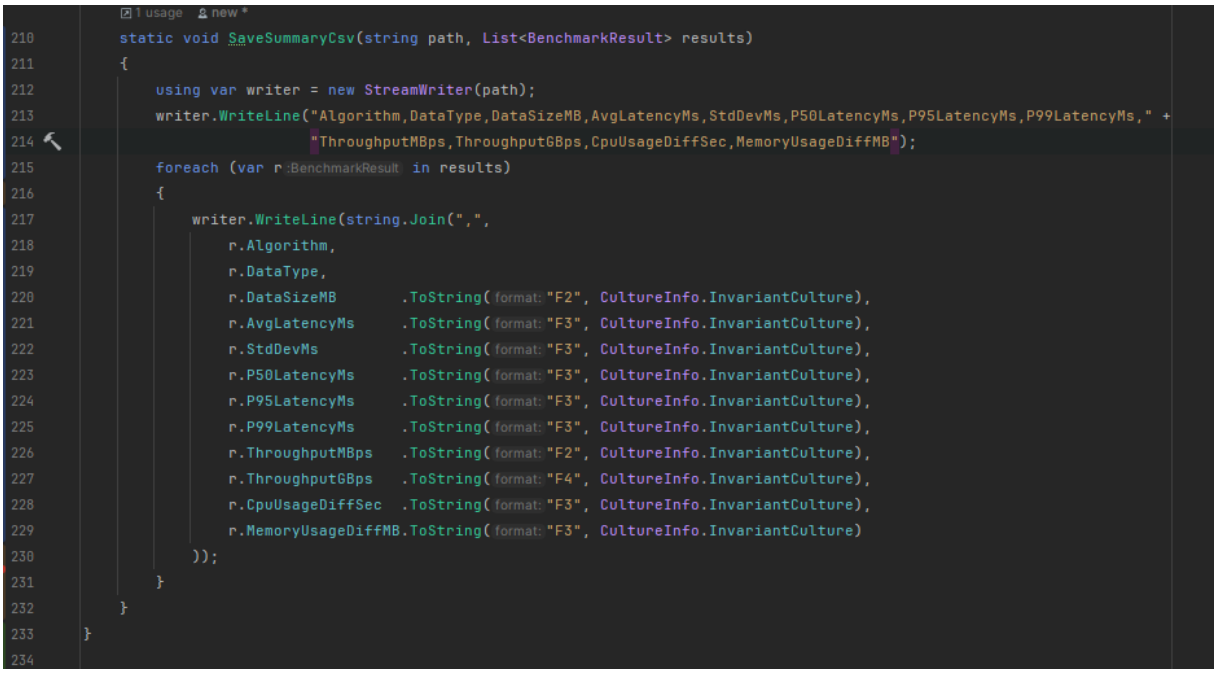
W celu zautomatyzowania pomiarów i zapewnienia ich powtarzalności, opracowano dedykowany kod w języku C#, wykorzystujący platformę .NET 8.0. Poniżej przedstawiono dwa kluczowe fragmenty implementacji odpowiedzialne za obliczanie metryk wydajnościowych oraz zapis wyników testów do pliku CSV.

Pomiar metryk w czasie iteracji.

```
87          for (int i = 0; i < Iterations; i++)
88          {
89              // Pomiar per-iteracja
90              TimeSpan cpuStart = Process.GetCurrentProcess().TotalProcessorTime;
91              long memStart     = GC.GetTotalMemory( forceFullCollection: false);
92
93              var sw = Stopwatch.StartNew();
94              _ = hashFunc(data);
95              sw.Stop();
96
97              TimeSpan cpuEnd = Process.GetCurrentProcess().TotalProcessorTime;
98              long memEnd     = GC.GetTotalMemory( forceFullCollection: false);
99
100             double latencyMs   = sw.Elapsed.TotalMilliseconds;
101             double cpuSec      = (cpuEnd - cpuStart).TotalSeconds;
102             double memMB       = (memEnd - memStart) / 1024.0 / 1024.0;
103             double throughput  = mb / sw.Elapsed.TotalSeconds;
104
105             iterationRecords.Add(new IterationResult
106             {
107                 Iteration      = i + 1,
108                 LatencyMs      = latencyMs,
109                 CpuTimeSec     = cpuSec,
110                 MemoryUsageMB  = memMB,
111                 ThroughputMBps = throughput
112             });
113         }
```

Zmierzona wartość dla każdej metryki była przypisywana do obiektu IterationResult i dodawana do listy wyników, co umożliwiało późniejszą analizę rozkładu i statystyk.

Po zakończeniu wszystkich iteracji i obliczeniu statystyk agregujących (średnich, percentyli, odchyleń), dane były zapisywane do pliku CSV z podsumowaniem dla każdego algorytmu i zestawu danych.



```
1 usage   new *
static void SaveSummaryCsv(string path, List<BenchmarkResult> results)
{
    using var writer = new StreamWriter(path);
    writer.WriteLine("Algorithm,DataType,DataSizeMB,AvgLatencyMs,StdDevMs,P50LatencyMs,P95LatencyMs,P99LatencyMs," +
                     "ThroughputMBps,ThroughputGBps,CpuUsageDiffSec,MemoryUsageDiffMB");
    foreach (var r :BenchmarkResult in results)
    {
        writer.WriteLine(string.Join(",",
            r.Algorithm,
            r.DataType,
            r.DataSizeMB       .ToString( format: "F2", CultureInfo.InvariantCulture),
            r.AvgLatencyMs     .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.StdDevMs         .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.P50LatencyMs     .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.P95LatencyMs     .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.P99LatencyMs     .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.ThroughputMBps   .ToString( format: "F2", CultureInfo.InvariantCulture),
            r.ThroughputGBps   .ToString( format: "F4", CultureInfo.InvariantCulture),
            r.CpuUsageDiffSec  .ToString( format: "F3", CultureInfo.InvariantCulture),
            r.MemoryUsageDiffMB.ToString( format: "F3", CultureInfo.InvariantCulture)
        ));
    }
}
}
```

W pierwszym wierszu pliku zapisywano nagłówki kolumn, w tym:

- Algorytm, Typ Danych, Rozmiar Danych (MB),
- Średni czas wykonania (ms), Odchylenie standardowe czasu (ms), Opóźnienie P50 (ms), Opóźnienie P95 (ms), Opóźnienie P99 (ms),
- Przepustowość (MB/s), Przepustowość (GB/s),
- Zużycie CPU (s), Zużycie Pamięci (MB).

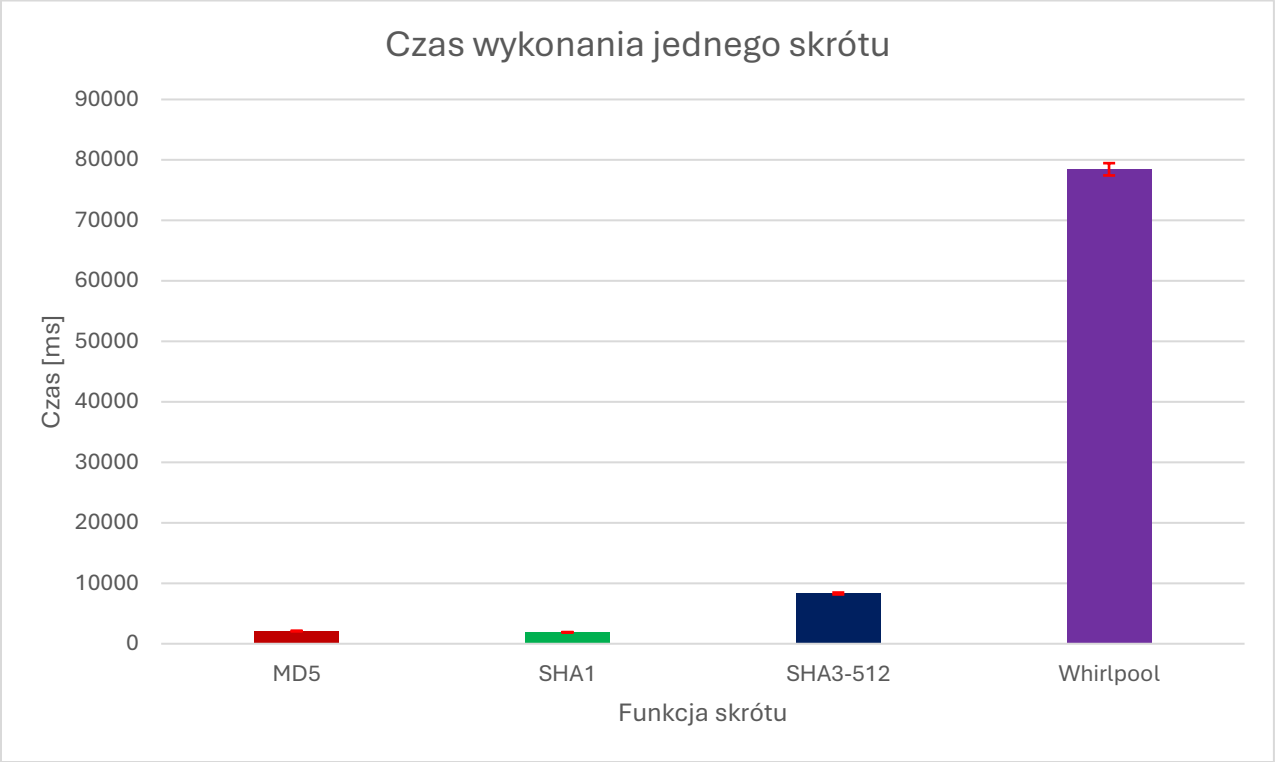
Dalsze wiersze zawierały sformatowane wyniki dla każdego przebiegu testowego. Użycie kultury InvariantCulture oraz precyzyjnych formatów (np. "F3", "F4") zapewniało spójność zapisu liczbowego niezależnie od ustawień regionalnych systemu.

Zaprojektowana architektura kodu umożliwiła dokładne rejestrowanie wyników pomiarów i stworzenie bazy danych testowych, która stanowiła podstawę do dalszej analizy przedstawionej w kolejnym rozdziale.

4. Wyniki i analiza danych

4.1. Plik ISO

W pierwszym teście porównano wydajność funkcji skrótu dla dużego obrazu systemu operacyjnego o rozmiarze 1179 MB. Celem było sprawdzenie, jak poszczególne algorytmy radzą sobie z przetwarzaniem dużych porcji danych pod względem czasu działania, stabilności, zużycia zasobów oraz przepustowości.



Rys. 1. Czas wykonania

Z wykresu wynika, że najniższy czas obliczeń osiągnęły funkcje MD5 (2107 ms) oraz SHA-1 (1904 ms). SHA3-512 był znacznie wolniejszy (8310 ms), natomiast funkcja Whirlpool wykazała skrajnie wysoki czas wykonania – ponad 78 s na jeden skrót, co czyni ją najmniej efektywną spośród analizowanych.

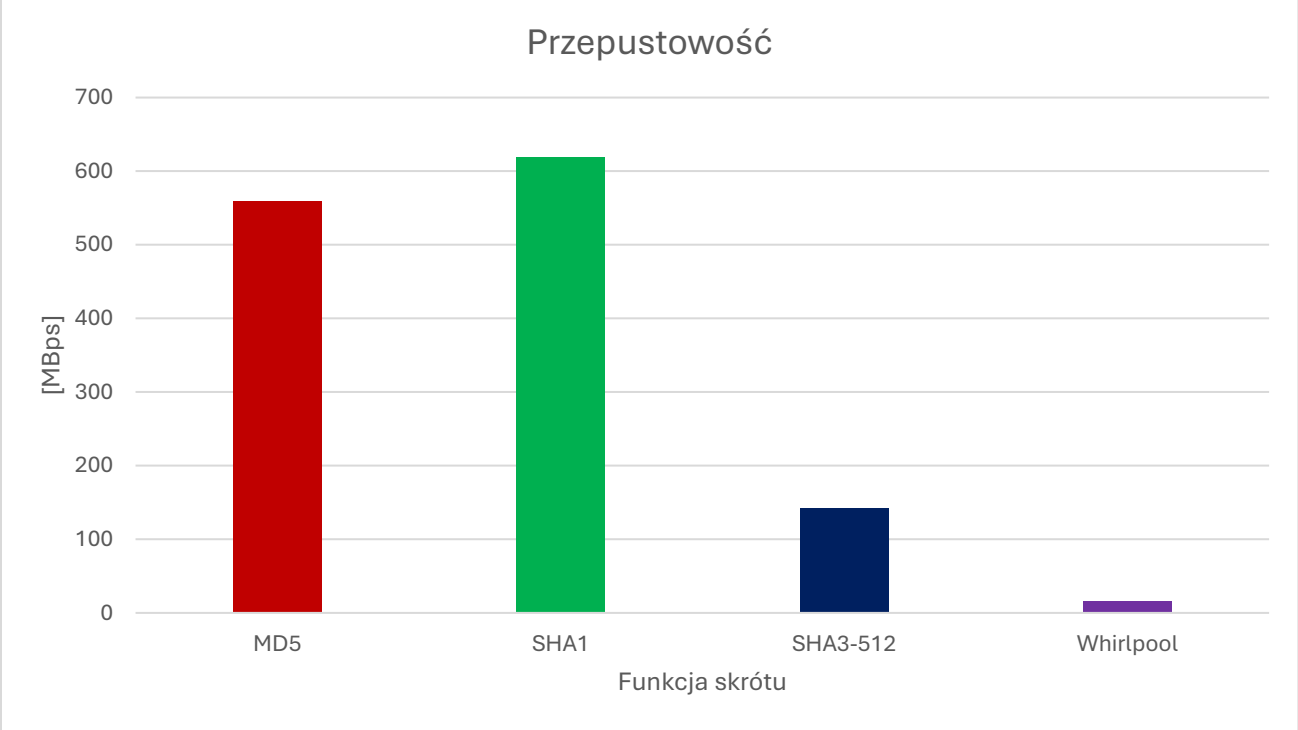
Pod względem stabilności (odchylenie standardowe i percentyle), SHA-1 wykazał najmniejsze wahania czasowe (StdDev: 28 ms), a także korzystne wartości percentyli:

- P50 (mediana): 1890 ms,
- P95: 1947 ms,
- P99: 1998 ms.

Dla porównania, Whirlpool miał bardzo wysokie wartości percentyli (np. P99 ≈ 82 259 ms), co oznacza dużą niestabilność w czasie działania.

Tab. 1. Zestawienie wyników dla pliku iso

Algorytm	Typ pliku	Rozmiar[MB]	Latencja[ms]	Odchylenie standardowe czasu wykonania[ms]	Latencja P50 [ms]	Latencja P95 [ms]	Latencja P99 [ms]
MD5	archlinux.iso	1179,03	2106,987	36,428	2092,155	2161,001	2203,182
SHA1	archlinux.iso	1179,03	1904,664	28,687	1890,75	1947,086	1998,261
SHA3-512	archlinux.iso	1179,03	8310,634	159,572	8272,825	8639,491	8846,098
Whirlpool	archlinux.iso	1179,03	78451,03	1014,602	78374,273	80679,142	82259,474



Rys. 2. Przepustowość (Throughput)

Pod względem przepustowości, najlepsze wyniki osiągnęły:

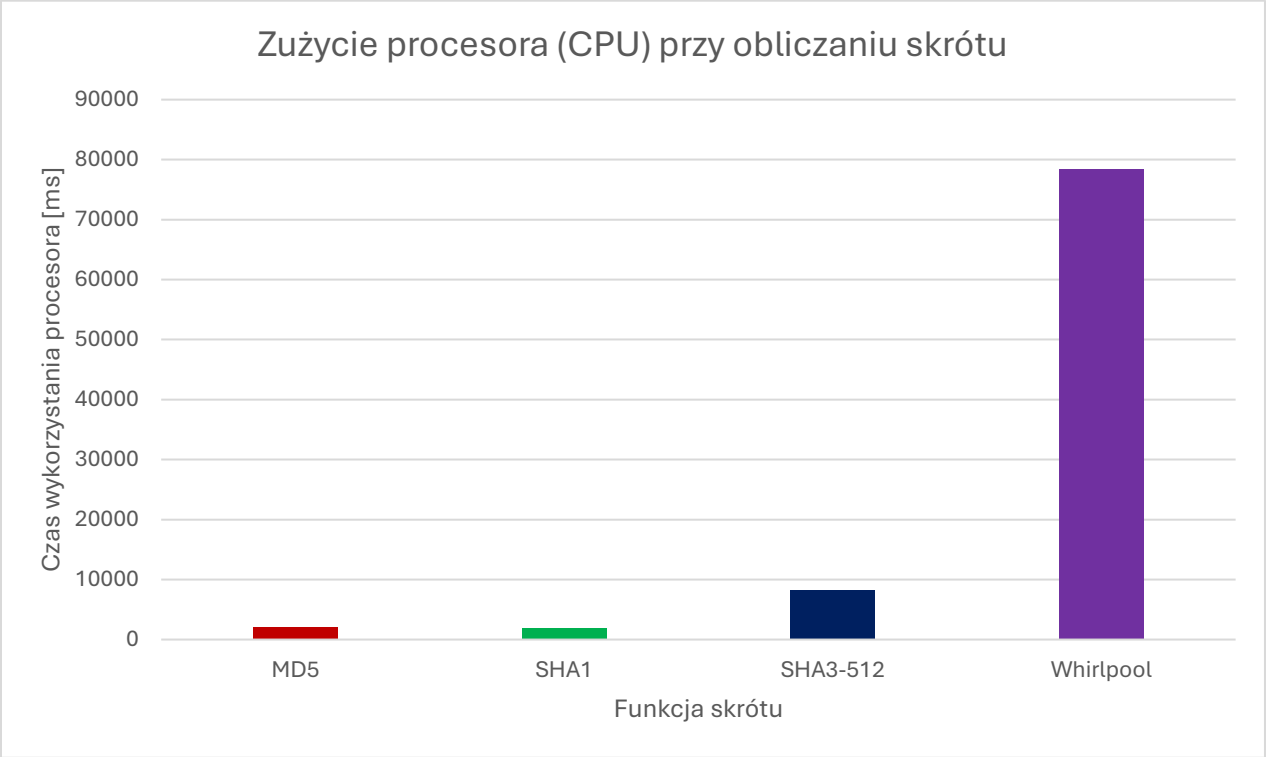
- SHA-1 – 619 MB/s,
- MD5 – 560 MB/s.

Warto zauważyć, że SHA3-512 osiągnął tylko 142 MB/s, a Whirlpool zaledwie 15 MB/s, co potwierdza jego niską efektywność przy dużych plikach.

Tab. 2. Zestawienie wyników przepustowości

Algorytm	Typ pliku	Rozmiar[MB]	Przepustowość[MB/s]
MD5	archlinux-2025.04.01-x86_64.iso	1179,03	559,58
SHA1	archlinux-2025.04.01-x86_64.iso	1179,03	619,02
SHA3-512	archlinux-2025.04.01-x86_64.iso	1179,03	141,87

Whirlpool	archlinux-2025.04.01-x86_64.iso	1179,03	15,03



Rys. 3. Zużycie CPU

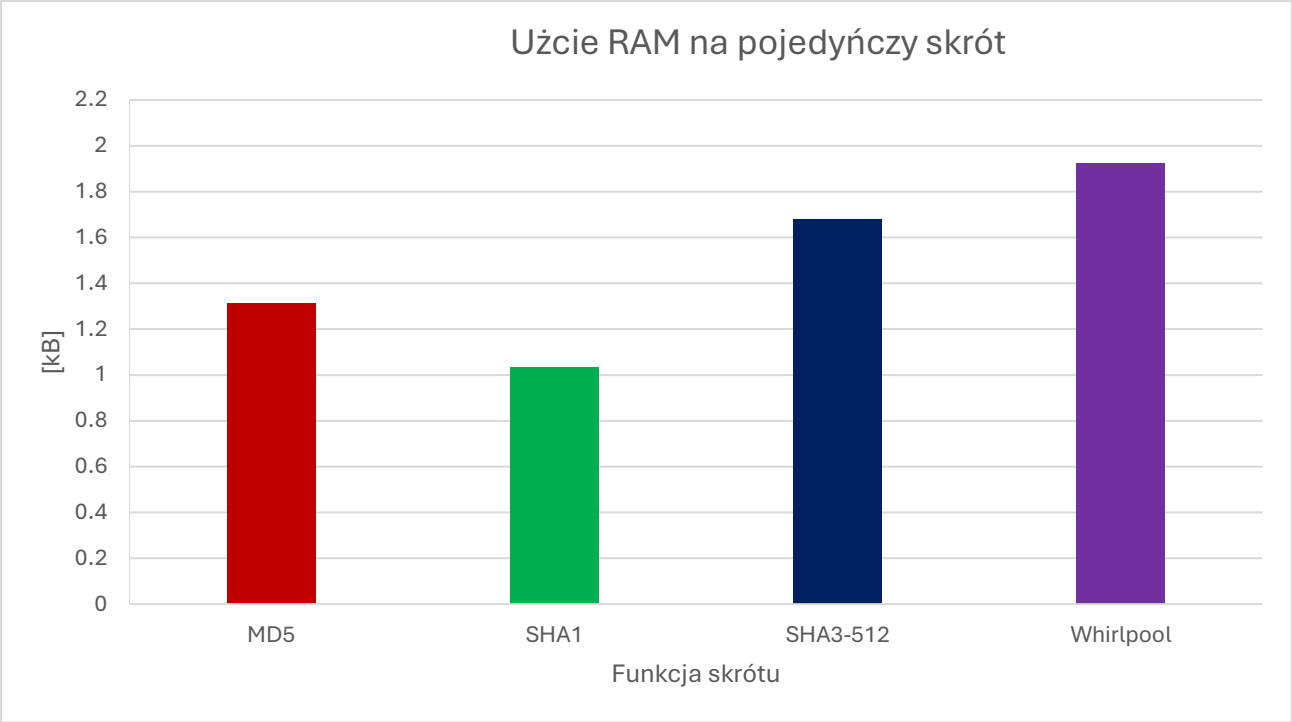
Na wykresie widoczne jest, że MD5 i SHA-1 zużyły bardzo niewielką ilość czasu procesora:

- MD5: 2,1 s,
- SHA-1: 1,9 s.

SHA3-512 zużył ponad 8 s, natomiast Whirlpool obciążył CPU przez aż 78 s, co koreluje z jego długim czasem przetwarzania.

Tab. 3. Zestawienie wyników CPU

Algorytm	Typ pliku	Rozmiar[MB]	Zużycie CPU (s)
MD5	archlinux-2025.04.01-x86_64.iso	1179,03	2090,94
SHA1	archlinux-2025.04.01-x86_64.iso	1179,03	1899,22
SHA3-512	archlinux-2025.04.01-x86_64.iso	1179,03	8269,84
Whirlpool	archlinux-2025.04.01-x86_64.iso	1179,03	78401,41



Rys. 4. Zużycie RAM

Podobnie w zakresie pamięci operacyjnej, wartości były relatywnie niskie, z niewielkimi różnicami:

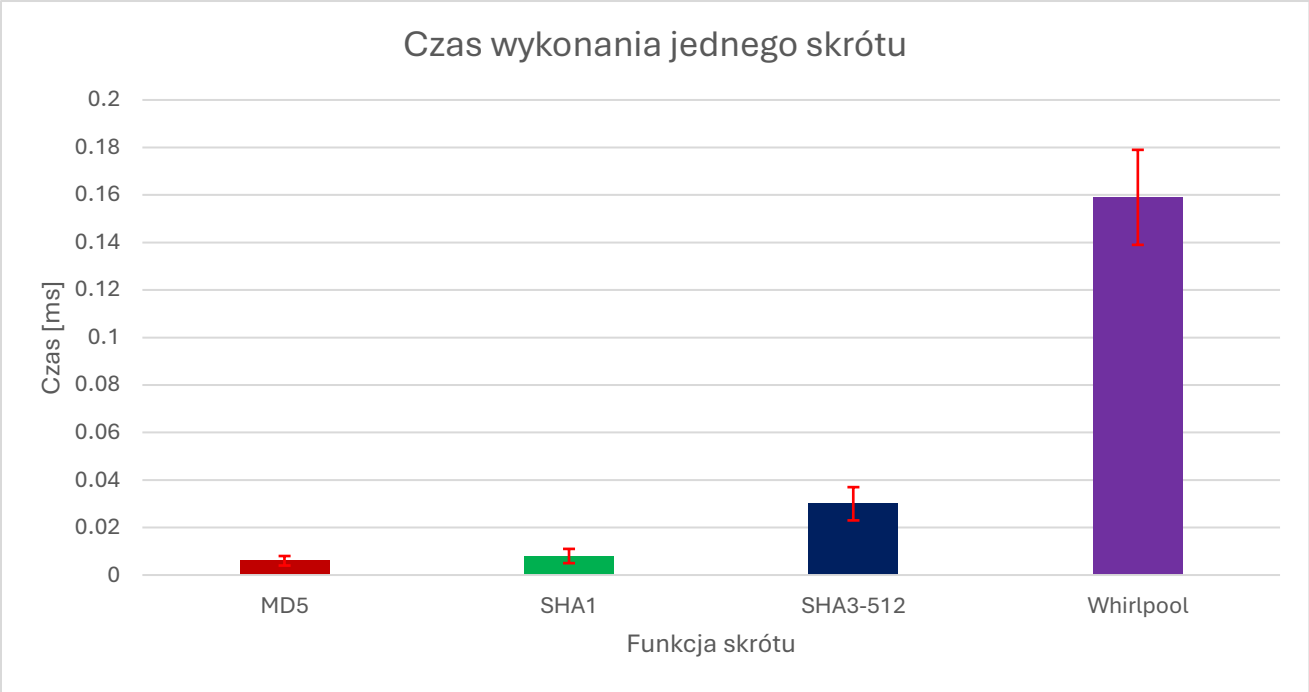
- SHA-1 – 1,03 MB,
- MD5 – 1,31 MB,
- SHA3-512 – 1,67 MB,
- Whirlpool – 1,93 MB.

Tab. 4. Zestawienie wyników RAM

Algorytm	Typ pliku	Rozmiar[MB]	Zużycie CPU (s)
MD5	archlinux-2025.04.01-x86_64.iso	1179,03	1,31072
SHA1	archlinux-2025.04.01-x86_64.iso	1179,03	1,03424
SHA3-512	archlinux-2025.04.01-x86_64.iso	1179,03	1,67936
Whirlpool	archlinux-2025.04.01-x86_64.iso	1179,03	1,92512

4.2. Certyfikat

W tej części badania przetestowano funkcje skrótu na bardzo małym pliku binarnym – 3kB typu certyfikat. Głównym celem było sprawdzenie, jak algorytmy radzą sobie z krótkimi danymi, co ma znaczenie m.in. w systemach uwierzytelniania, podpisów cyfrowych i protokołach sieciowych.



Rys. 5. Czas wykonania

Jak pokazuje wykres, wszystkie algorytmy uzyskały bardzo niskie wartości latencji, mierzone w milisekundach:

- MD5: 0,006 ms,
- SHA-1: 0,008 ms,
- SHA3-512: 0,037 ms,
- Whirlpool: 0,159 ms.

Pomimo ogólnej szybkości, Whirlpool wypada zauważalnie wolniej niż pozostałe funkcje. Najszybszy okazał się MD5, tuż za nim SHA-1.

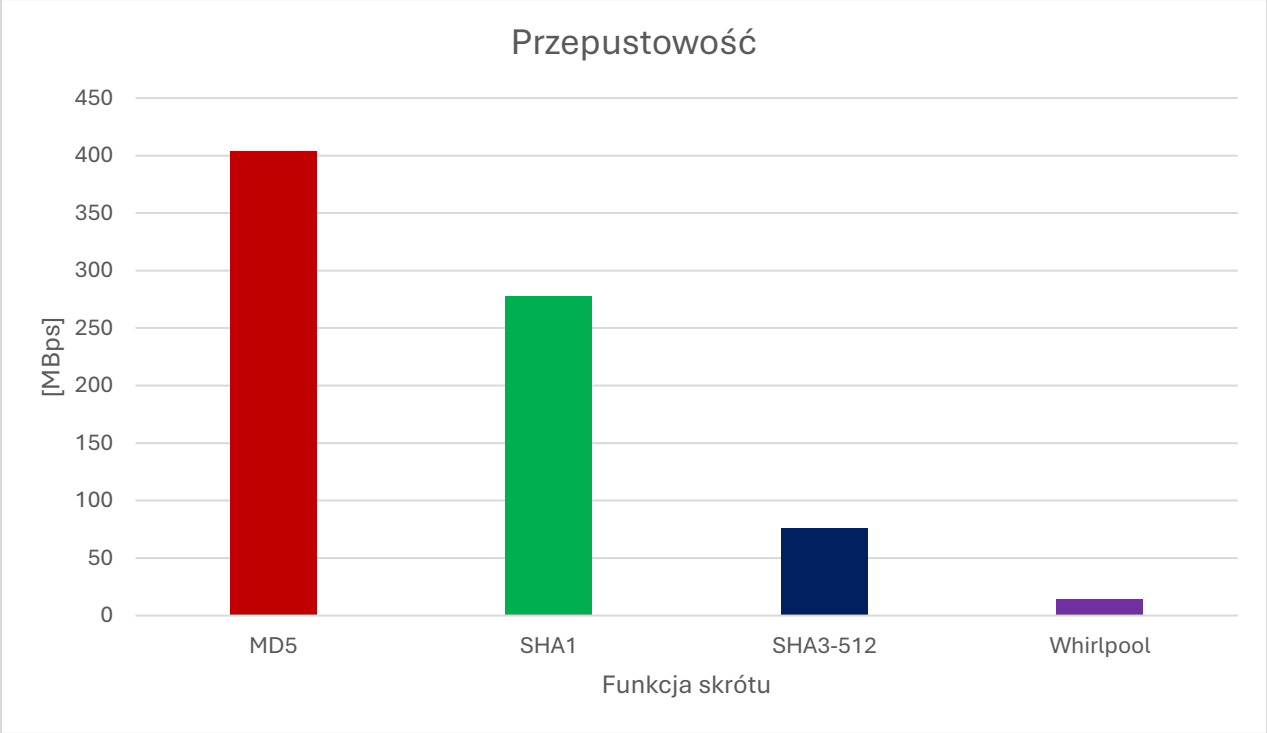
Pod względem stabilności (odchylenie standardowe i percentyle), wszystkie algorytmy osiągnęły bardzo niskie wartości, co wskazuje na dużą powtarzalność wyników:

- MD5: StdDev 0,002 ms; P99 = 0,018 ms,
- SHA-1: StdDev 0,003 ms; P99 = 0,021 ms,
- SHA3-512: StdDev 0,007 ms; P99 = 0,036 ms,
- Whirlpool: StdDev 0,020 ms; P99 = 0,220 ms.

Tab. 5. Zestawienie wyników dla certyfikatu

Algorytm	Typ pliku	Rozmiar danych [MB]	Czas wykonania [ms]	Odchylenie standardowe czasu (ms)[ms]	Latencja P50	Latencja P95	Latencja P99
MD5	_.facebookcom.crt	0	0,006	0,002	0,005	0,006	0,018
SHA1	_.facebookcom.crt	0	0,008	0,003	0,008	0,016	0,021

SHA 3-512	_.facebookcom.crt	0	0,03	0,007	0,029	0,043	0,058
Whirlpool	_.facebookcom.crt	0	0,159	0,02	0,152	0,209	0,22



Rys. 6. Przepustowość

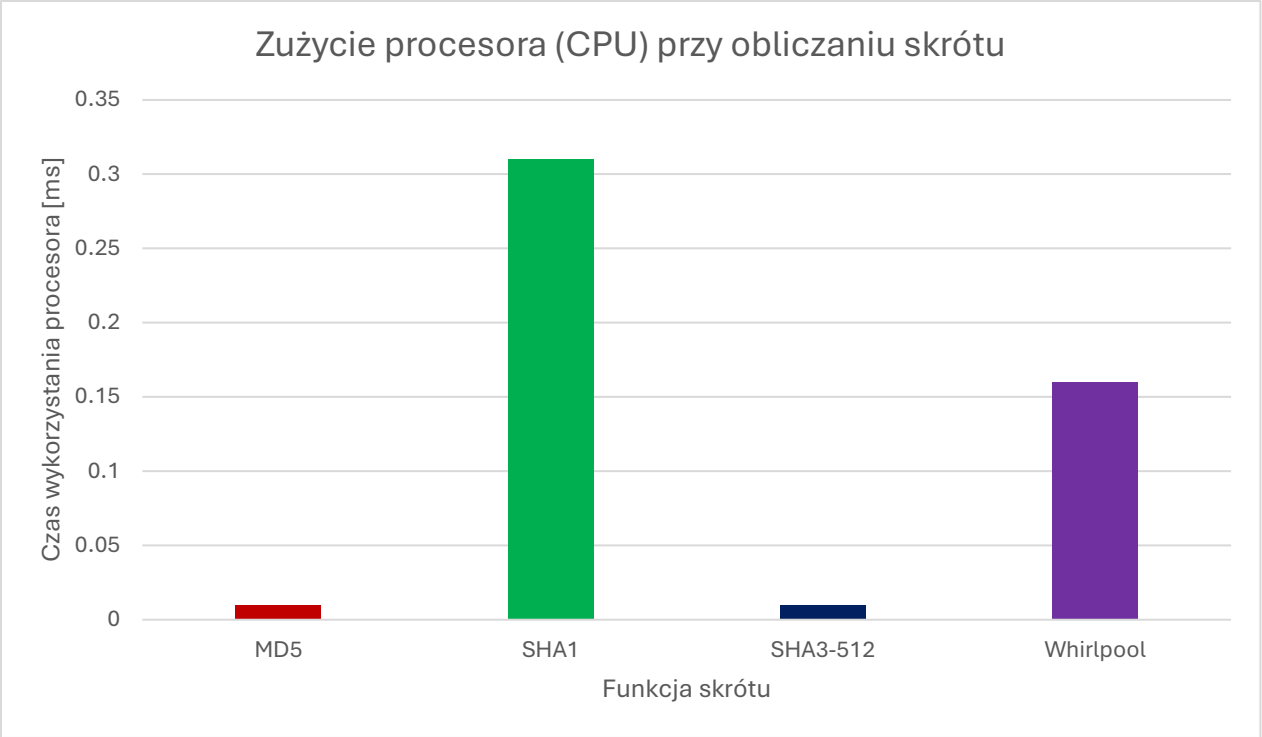
Dla tak małego pliku względna przepustowość staje się mniej reprezentatywna, jednak nadal pozwala zauważyć różnice:

- MD5 osiągnął najlepszy wynik: 404 MB/s,
- SHA-1 – 278 MB/s,
- SHA3-512 – 76 MB/s,
- Whirlpool – zaledwie 14 MB/s.

Przy małych danych SHA3-512 i Whirlpool wyraźnie ustępują prostszym i lżejszym funkcjom.

Tab. 6. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Przepustowość (MB/s)
MD5	_.facebookcom.crt	404,12
SHA1	_.facebookcom.crt	277,94
SHA3-512	_.facebookcom.crt	75,89
Whirlpool	_.facebookcom.crt	14,29



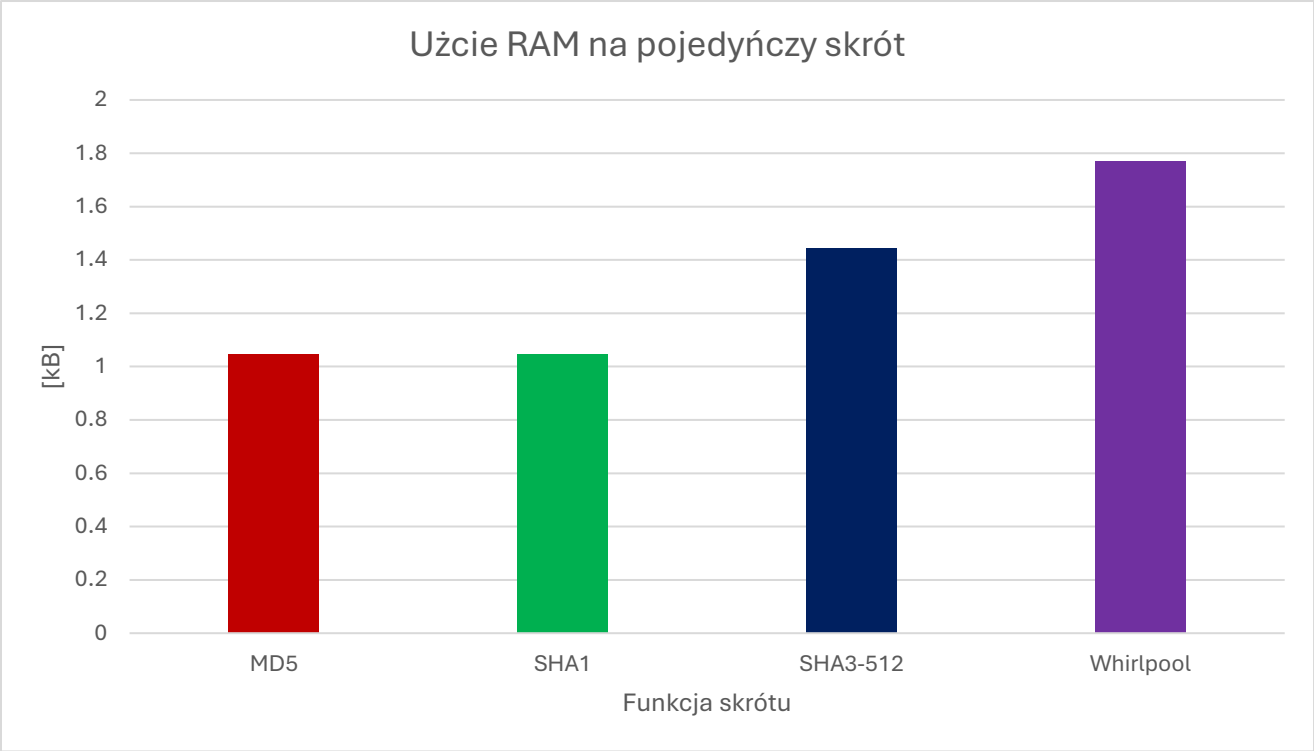
Rys. 7. Zużycie CPU

W kontekście czasu użycia procesora:

- Najlepiej wypadły MD5 (0,01 s) i SHA3-512 (0,01 s),
- SHA-1 miał wyraźnie wyższy czas: 0,31 s,
- Whirlpool: 0,16 s.

Tab. 7. Zestawienie wyników CPU

Algorytm	Typ Danych	Zużycie CPU (ms)
MD5	_.facebookcom.crt	0,01
SHA1	_.facebookcom.crt	0,31
SHA3-512	_.facebookcom.crt	0,01
Whirlpool	_.facebookcom.crt	0,16



Rys. 8. Zużycie RAM

W zakresie pamięci RAM, różnice nie były duże, ale widoczne:

- SHA-1 – 1,04 MB,
- MD5 – 1,04 MB,
- SHA3-512 – 1,43 MB,
- Whirlpool – 1,77 MB.

Tab. 8. Zestawienie wyników RAM

Algorytm	Typ Danych	Zużycie Pamięci (kB)
MD5	_.facebookcom.crt	1,04448
SHA1	_.facebookcom.crt	1,04448
SHA3-512	_.facebookcom.crt	1,44384
Whirlpool	_.facebookcom.crt	1,77152

4.3. Dokument PDF

W trzecim przypadku porównano wydajność funkcji skrótu podczas przetwarzania średniej wielkości dokumentu PDF o rozmiarze 4,12 MB. Tego rodzaju pliki są powszechnie wykorzystywane w kontekście archiwizacji, podpisów cyfrowych i systemów dokumentacji elektronicznej.

Rys. 9. Czas wykonania

Na podstawie wykresu widać, że funkcje MD5 i SHA-1 są zdecydowanie najszybsze, z czasem wykonania rzędu:

- MD5: 7,21 ms,
- SHA-1: 6,62 ms.

Nieco dłużej trwało wykonanie funkcji SHA3-512 – 28,58 ms, natomiast Whirlpool ponownie wypadł znacznie wolniej – 270,96 ms, czyli ponad 9 razy wolniej niż SHA3-512 i ponad 37 razy wolniej niż SHA-1.

Pod względem stabilności:

- SHA-1 i MD5 wykazały spójne działanie z bardzo niskim odchyleniem standardowym (0,66 ms i 0,13 ms),
- SHA3-512 i Whirlpool były mniej stabilne – odpowiednio 1,91 ms oraz 5,79 ms,
- Percentyle P99 potwierdzają te różnice:
  - MD5: 7,47 ms,
  - SHA-1: 6,84 ms,
  - SHA3-512: 34,1 ms,
  - Whirlpool: aż 291,5 ms.

Tab. 9. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Średni czas wykonania (ms)	Odchylenie standardowe czasu (ms)	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	DTU-1-Wprowadzenie.pdf	4,12	7,205	0,134	7,176	7,426	7,618

SHA 1	DTU-1-Wprowadzenie.pdf	4,12	6,618	0,657	6,517	6,847	8,375
SHA 3-512	DTU-1-Wprowadzenie.pdf	4,12	28,582	1,91	28,061	31,454	37,644
Whirlpool	DTU-1-Wprowadzenie.pdf	4,12	270,957	5,79	269,796	276,311	291,51

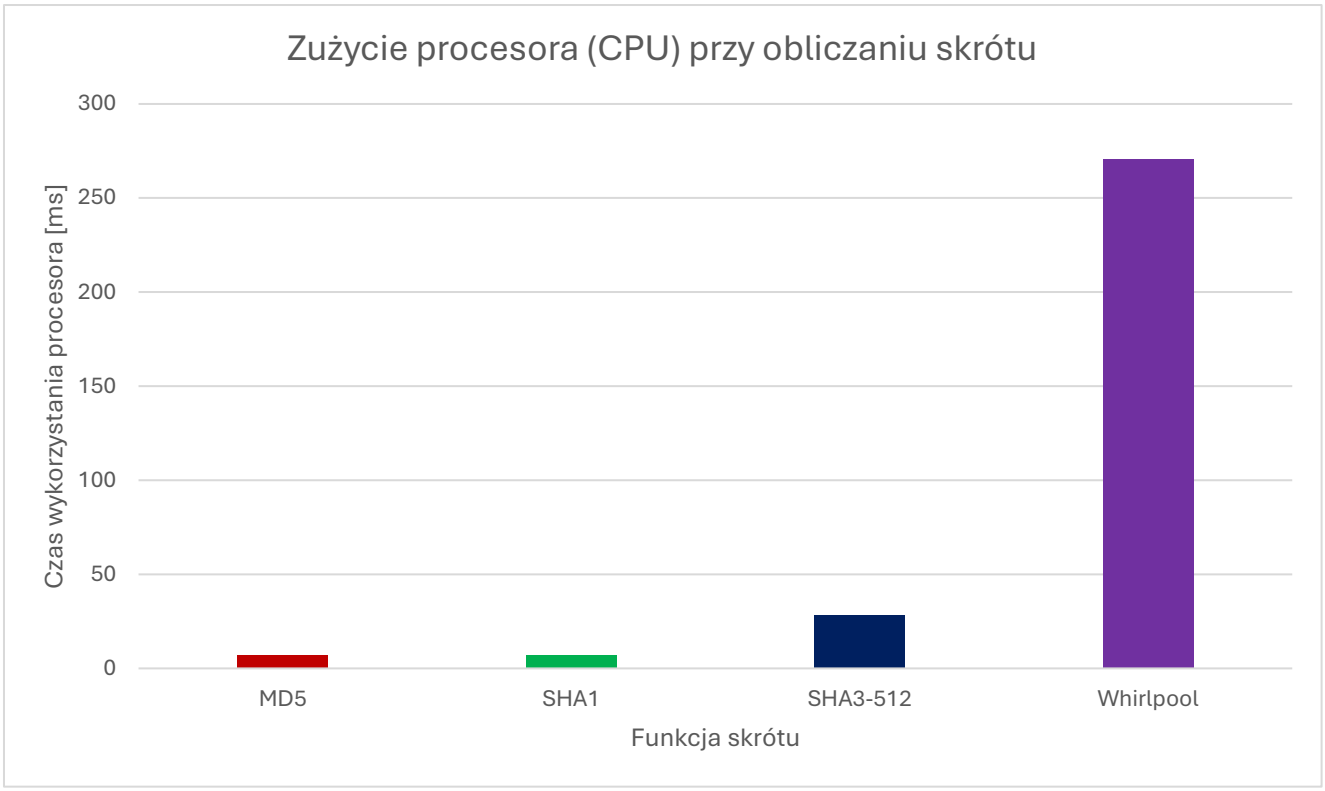
Rys. 10. Przepustowość

Z wykresu „Przepustowość" wynika, że:

- Najwyższe wartości osiągnęły SHA-1 (622 MB/s) oraz MD5 (571 MB/s),
- SHA3-512 przetwarzał dane z prędkością 144 MB/s,
- Whirlpool – jedynie 15 MB/s, co potwierdza jego niską efektywność przy plikach średniej wielkości.

Tab. 10. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	DTU-1-Wprowadzenie.pdf	4,12	571,37
SHA1	DTU-1-Wprowadzenie.pdf	4,12	622,08
SHA3-512	DTU-1-Wprowadzenie.pdf	4,12	144,03
Whirlpool	DTU-1-Wprowadzenie.pdf	4,12	15,19



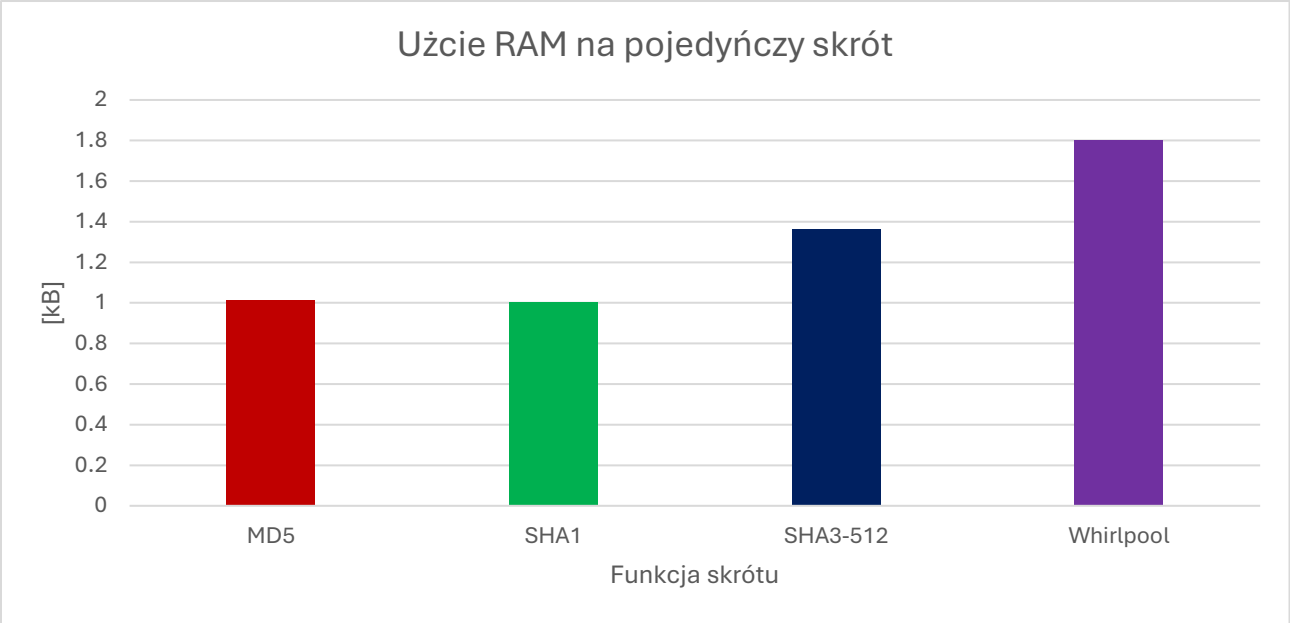
Rys. 11. Zużycie CPU

Pod względem zużycia CPU:

- SHA-1 był najefektywniejszy – 0,69 s czasu procesora,
- MD5 zużył nieco więcej – 0,72 s,
- SHA3-512 – 2,86 s,
- Whirlpool – ponad 27 s, co jest znacząco wyższe niż pozostałe.

Tab. 11. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)
MD5	DTU-1-Wprowadzenie.pdf	4,12	7,19
SHA1	DTU-1-Wprowadzenie.pdf	4,12	6,88
SHA3-512	DTU-1-Wprowadzenie.pdf	4,12	28,59
Whirlpool	DTU-1-Wprowadzenie.pdf	4,12	270,78



Rys. 12. Zużycie RAM

W zakresie pamięci RAM różnice były nieznaczne:

- SHA-1 – 1,00 MB,
- MD5 – 1,01 MB,
- SHA3-512 – 1,36 MB,
- Whirlpool – 1,80 MB.

Tab. 12. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	DTU-1-Wprowadzenie.pdf	4,12	1,01376
SHA1	DTU-1-Wprowadzenie.pdf	4,12	1,00352
SHA3-512	DTU-1-Wprowadzenie.pdf	4,12	1,36192
Whirlpool	DTU-1-Wprowadzenie.pdf	4,12	1,80224

4.4. Archiwum TAR obrazu docker

W tej części analizy sprawdzono wydajność funkcji skrótu podczas przetwarzania obrazu Dockera postgresfile.tar o rozmiarze 149,09 MB. Takie pliki są typowe w środowiskach CI/CD, konteneryzacji i automatyzacji infrastruktury – dlatego istotne jest, by operacje haszujące były wydajne i stabilne.

Rys. 13. Czas wykonania

Na wykresie „Czas wykonania jednego skrótu" widzimy wyraźną przewagę prostych funkcji:

- SHA-1: 235,38 ms,
- MD5: 250,63 ms,
- SHA3-512: 1017,57 ms,
- Whirlpool: aż 9867,47 ms (~10 s).

Stabilność czasowa mierzona odchyleniem standardowym również przemawia za SHA-1 i MD5:

- SHA-1: 4,56 ms,
- MD5: 7,20 ms.

SHA3-512 i Whirlpool miały znacznie większe wahania – odpowiednio 18,36 ms i aż 129,08 ms, co potwierdzają także wysokie percentyle:

- Whirlpool P99: 10 636 ms,
- SHA3-512 P99: 1 059 ms.

Tab. 13. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Średni czas wykonania (ms)	Odchylenie standardowe czasu (ms)	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	postgresfile.tar	149,09	260,632	1,703	260,163	263,644	266,646
SHA1	postgresfile.tar	149,09	235,379	4,556	234,365	242,067	248,81
SHA3-512	postgresfile.tar	149,09	1017,569	18,396	1011,289	1059,237	1092,535

Whirlpool	postgresfile.tar	149,09	9867,47	129,078	9825,21	10062,03	10346,71

Rys. 14. Przepustowość

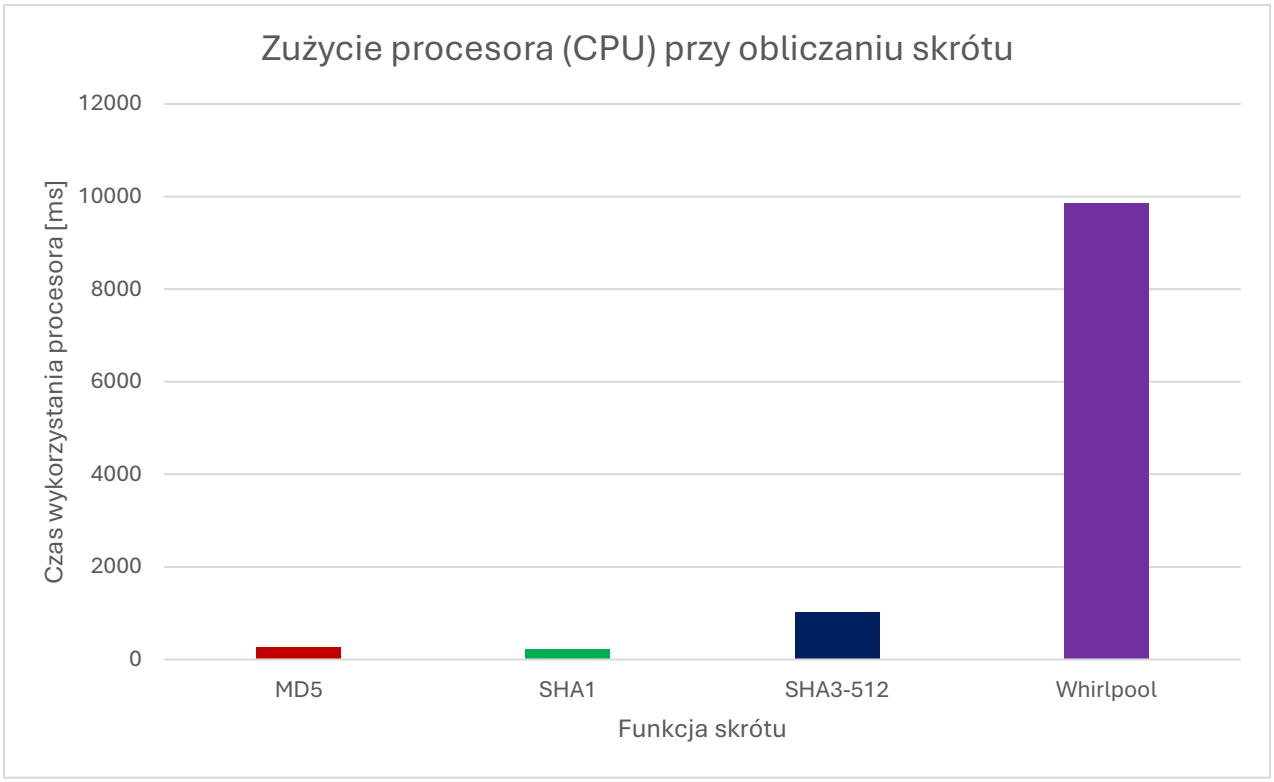
W kontekście przetwarzania dużych bloków danych:

- SHA-1 osiągnął najlepszy wynik: 624 MB/s,
- MD5 również wysoko – 589 MB/s,
- SHA3-512: tylko 146 MB/s,
- Whirlpool – bardzo nisko: 15 MB/s.

Widać wyraźnie, że nowocześniejsze i bardziej złożone funkcje skrótu (SHA3 i Whirlpool) wypadają znacznie gorzej przy większych plikach, jeśli chodzi o przepustowość.

Tab. 14. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	postgresfile.tar	149,09	572,03
SHA1	postgresfile.tar	149,09	633,4
SHA3-512	postgresfile.tar	149,09	146,52
Whirlpool	postgresfile.tar	149,09	15,11



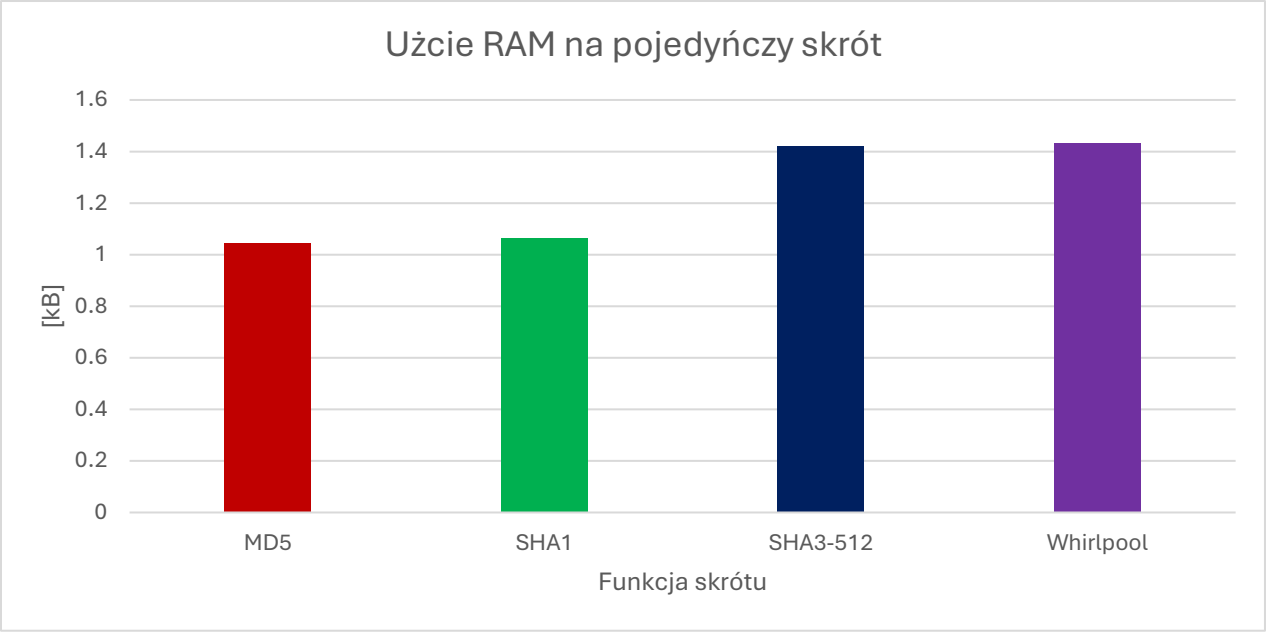
Rys. 15. Zużycie CPU

Na wykresie „Zużycie procesora" dominująca jest funkcja Whirlpool – zużyła aż 9860 s czasu CPU, co znacząco przekracza wartości dla pozostałych:

- SHA-1: 23,5 s,
- MD5: 26,1 s,
- SHA3-512: 101,8 s.

Tab. 15. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)
MD5	postgresfile.tar	149,09	260.62
SHA1	postgresfile.tar	149,09	235.31
SHA3-512	postgresfile.tar	149,09	1017.5
Whirlpool	postgresfile.tar	149,09	9860.78



Rys. 16. Zużycie RAM

W przypadku użycia pamięci RAM wartości były umiarkowane i stosunkowo zbliżone:

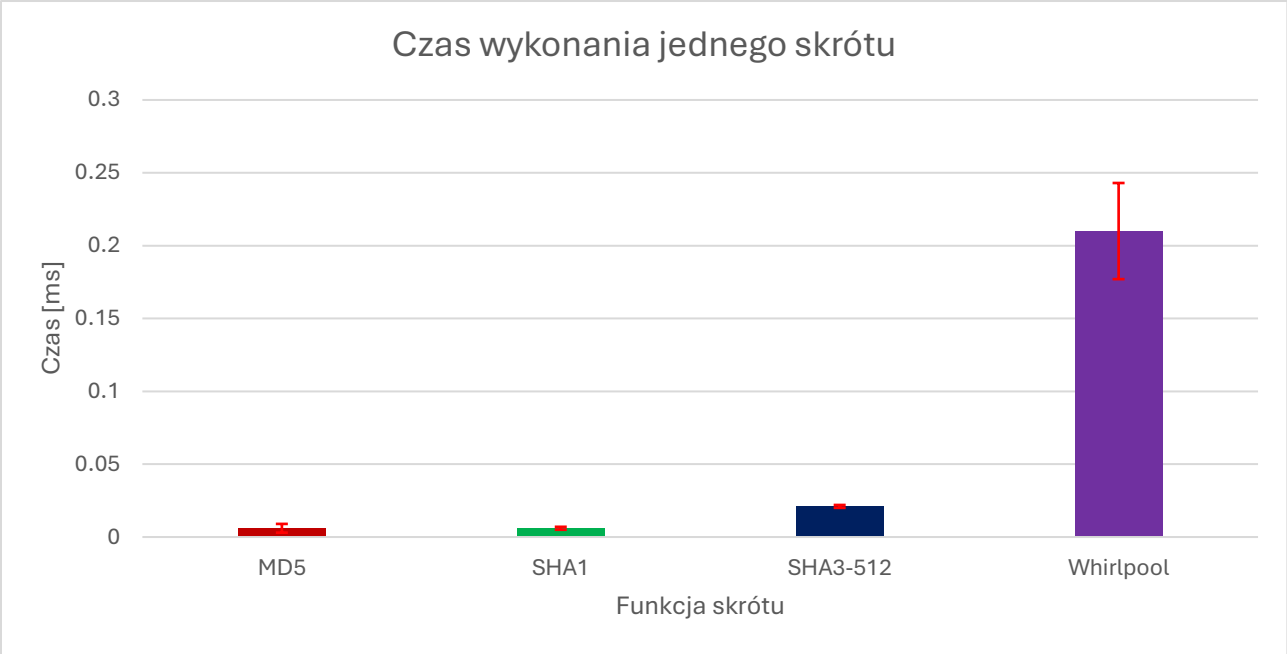
- SHA-1: 1,06 MB,
- MD5: 1,04 MB,
- SHA3-512: 1,42 MB,
- Whirlpool: 1,43 MB

Tab. 16. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	postgresfile.tar	149,09	1,04448
SHA1	postgresfile.tar	149,09	1,06496
SHA3-512	postgresfile.tar	149,09	1,42336
Whirlpool	postgresfile.tar	149,09	1,4336

4.5. Plik tekstowy wypełniony 0x00 3kB

W tej części eksperymentu zbadano zachowanie funkcji skrótu przy przetwarzaniu bardzo małego pliku tekstowego o rozmiarze **3 kB**, wypełnionego wzorcem 0x00. Takie dane mogą reprezentować różnego rodzaju małe rekordy, nagłówki czy segmenty danych systemowych.



Rys. 17. Czas wykonania

Zgodnie z wykresem „Czas wykonania jednego skrótu", latencje dla tego typu danych są ekstremalnie niskie, mierzone w tysięcznych częściach milisekundy:

- **MD5**: 0,006 ms,
- **SHA-1**: 0,006 ms,
- **SHA3-512**: 0,021 ms,
- **Whirlpool**: 0,198 ms.

Wszystkie algorytmy poza Whirlpool wykazały bardzo wysoką stabilność:

- StdDev dla MD5, SHA-1 i SHA3-512 mieściło się w zakresie 0,001–0,003 ms,
- Whirlpool wyróżniał się wysokim rozrzutem wyników: **0,033 ms**, co w tym kontekście stanowi dużą wartość.

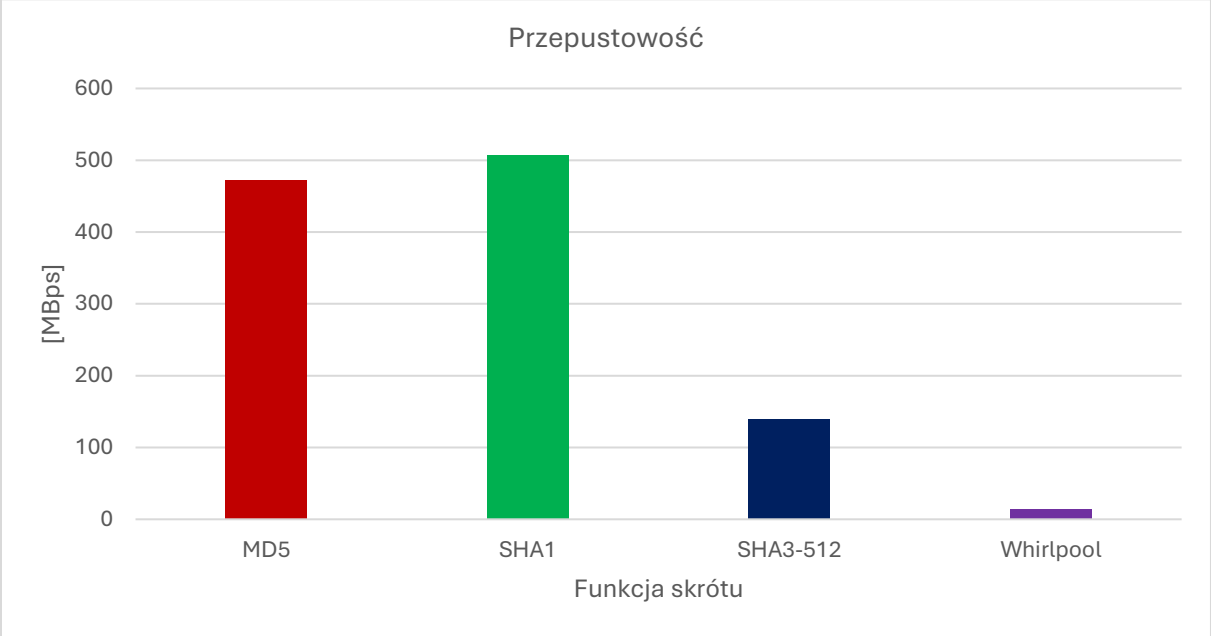
Percentyle P99 pokazują, że:

- MD5 i SHA-1 są niemal natychmiastowe (0,012 ms),
- SHA3-512: 0,026 ms,
- Whirlpool: 0,336 ms – prawie **30 razy wolniej** niż MD5 przy najwyższych wartościach.

Tab. 17. Zestawienie wyników latencji

Algorytm	Typ Danych	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	null3KB.txt	0,006	0,003	0,006	0,007	0,019
SHA1	null3KB.txt	0,006	0,001	0,005	0,007	0,012

SHA3-512	null3KB.txt	0,021	0,001	0,021	0,023	0,026
Whirlpool	null3KB.txt	0,21	0,033	0,198	0,292	0,336



Rys. 18. Przepustowość

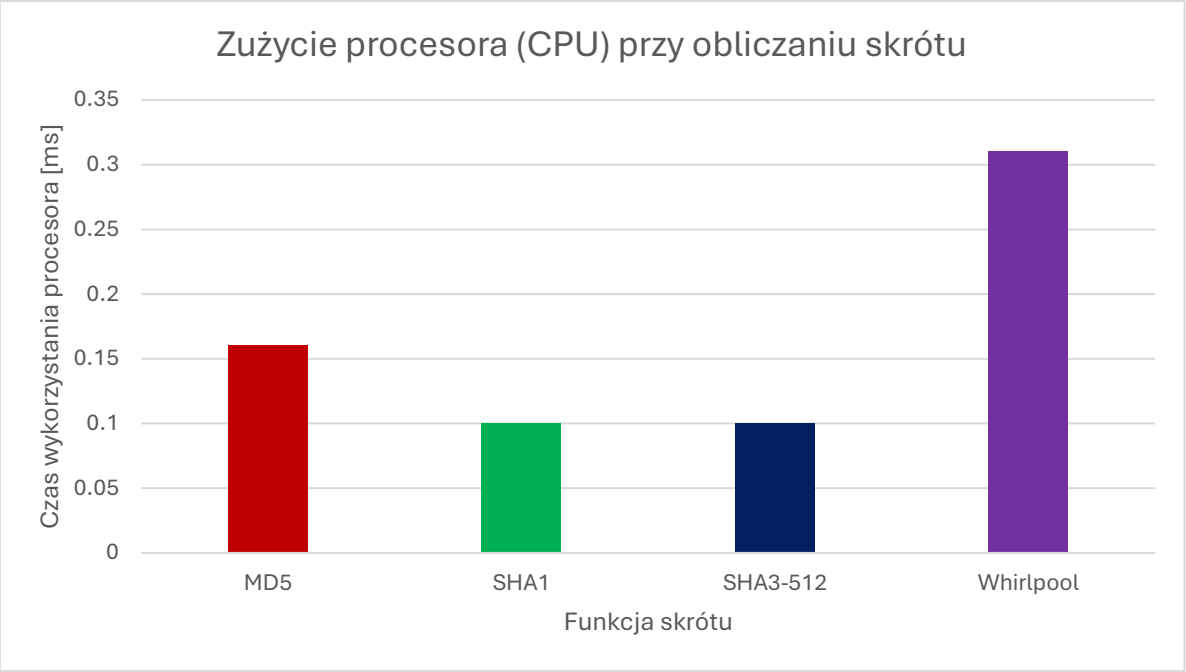
W odniesieniu do bardzo małych plików:

- **SHA-1** osiągnął największą przepustowość – **507 MB/s**,
- **MD5**: 472 MB/s,
- **SHA3-512**: 139 MB/s,
- **Whirlpool**: jedynie 13,9 MB/s.

Pomimo niewielkich danych wejściowych, różnice w konstrukcji algorytmu mają bezpośredni wpływ na efektywność.

Tab. 18. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Przepustowość (MB/s)
MD5	null3KB.txt	472,07
SHA1	null3KB.txt	506,95
SHA3-512	null3KB.txt	138,87
Whirlpool	null3KB.txt	13,92



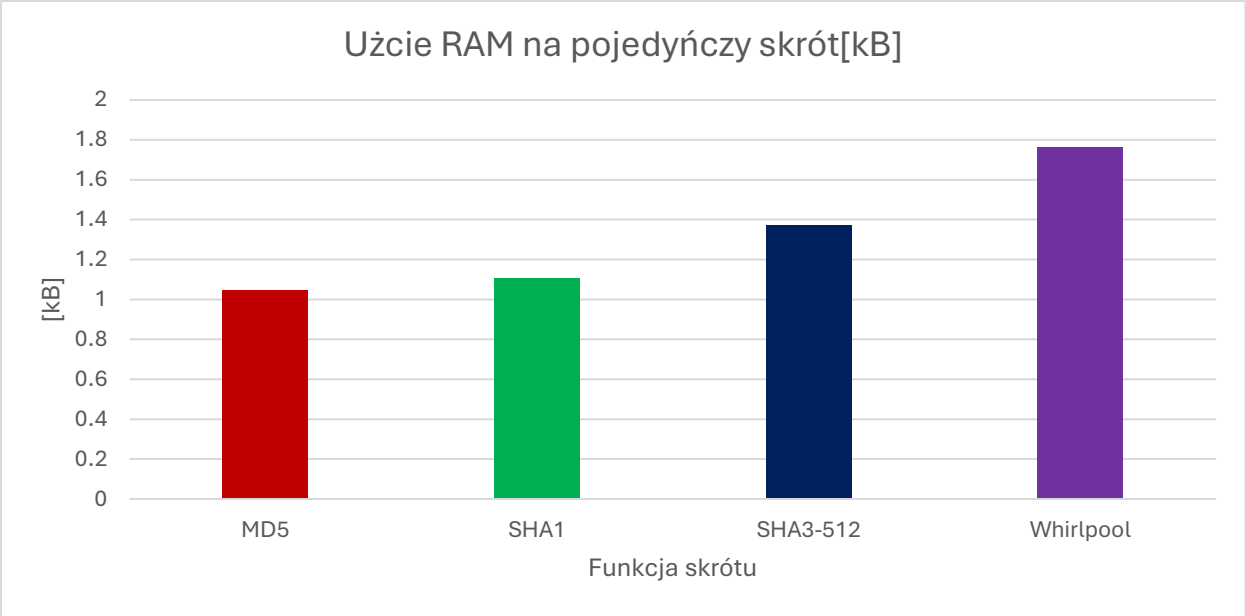
Rys. 19. Zużycie CPU

Wszystkie algorytmy cechowały się niskim zużyciem czasu procesora, rzędu ułamków milisekundy:

- **SHA-1**: 0,10 ms,
- **MD5**: 0,16 ms,
- **SHA3-512**: 0,10 ms,
- **Whirlpool**: 0,31 ms.

Tab. 19. Zestawienie wyników CPU

Algorytm	Typ Danych	Zużycie CPU (ms)
MD5	null3KB.txt	0.16
SHA1	null3KB.txt	0.1
SHA3-512	null3KB.txt	0.1
Whirlpool	null3KB.txt	0.31



Rys. 20. Zużycie RAM

Pod względem użycia pamięci RAM na pojedynczy skrót:

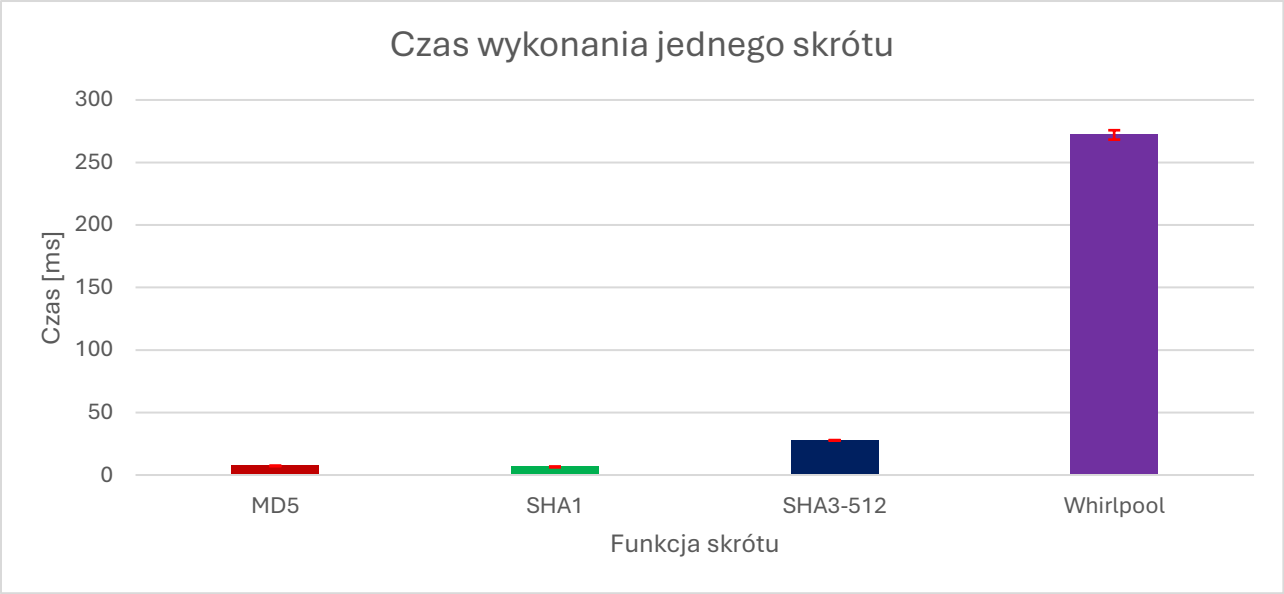
- **MD5**: 1,04 kB,
- **SHA-1**: 1,10 kB,
- **SHA3-512**: 1,37 kB,
- **Whirlpool**: 1,76 kB.

Tab. 20. Zestawienie wyników RAM

Algorytm	Typ Danych	Zużycie Pamięci (kB)
MD5	null3KB.txt	104,448
SHA1	null3KB.txt	110,592
SHA3-512	null3KB.txt	137,216
Whirlpool	null3KB.txt	176,128

4.6. Plik tekstowy wypełniony 0x00 4MB

Ostatni test dotyczył pliku tekstowego o rozmiarze **4,13 MB**, w całości wypełnionego bajtem 0x00. Tego typu dane syntetyczne pozwalają na ocenę, jak funkcje skrótu radzą sobie w przypadku jednolitych i powtarzalnych wzorców danych, co może mieć znaczenie w systemach bazujących na danych generowanych maszynowo lub testowo.



Rys. 21. Czas wykonania

Zgodnie z wykresem „Czas wykonania jednego skrótu", najniższe wartości latencji osiągnęły:

- **SHA-1**: 6,45 ms,
- **MD5**: 7,24 ms.
  Funkcja **SHA3-512** potrzebowała znacznie więcej czasu: **27,70 ms**, a **Whirlpool** ponownie wykazał wyraźnie najdłuższy czas: **272,01 ms**.

Pod względem stabilności czasowej:

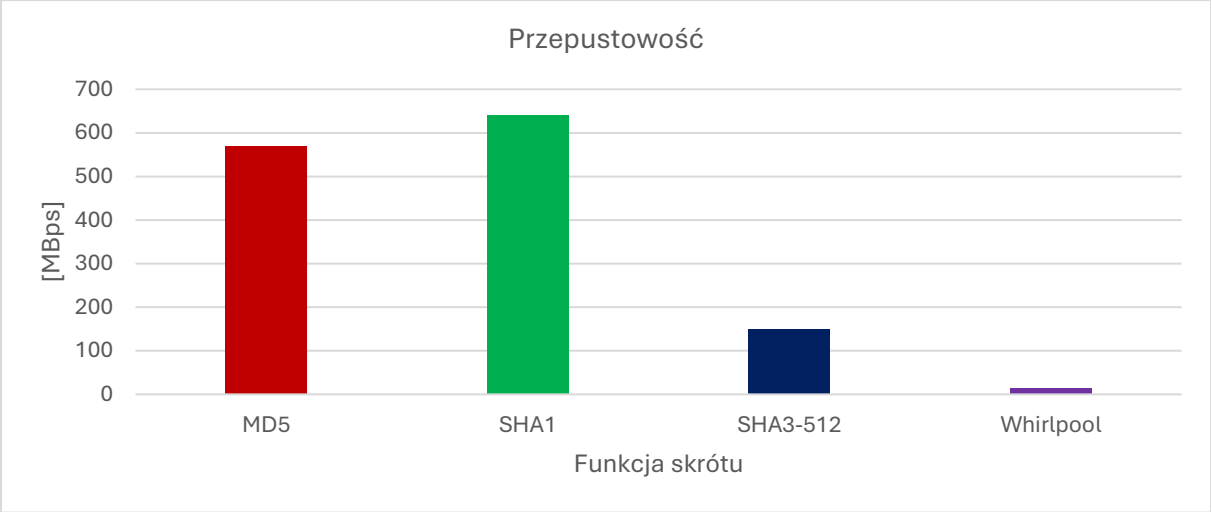
- **SHA-1** wyróżnia się niskim odchyleniem standardowym – 0,366 ms,
- **MD5**: 0,159 ms,
- **SHA3-512**: 0,224 ms,
- **Whirlpool**: 3,76 ms, co w tym przypadku jest znaczące.

Percentyle P99 dodatkowo potwierdzają spójność SHA-1 (6,93 ms) i wysoką niestabilność Whirlpoola (290,85 ms).

Tab. 21. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	null4216KB.txt	4,13	7,24	0,159	7,228	7,509	7,809
SHA1	null4216KB.txt	4,13	6,454	0,366	6,39	6,674	6,934
SHA3-512	null4216KB.txt	4,13	27,703	0,224	27,661	28,231	28,472

Whirlpool	null4216KB.txt	4,13	272,014	3,762	270,895	278,761	290,853



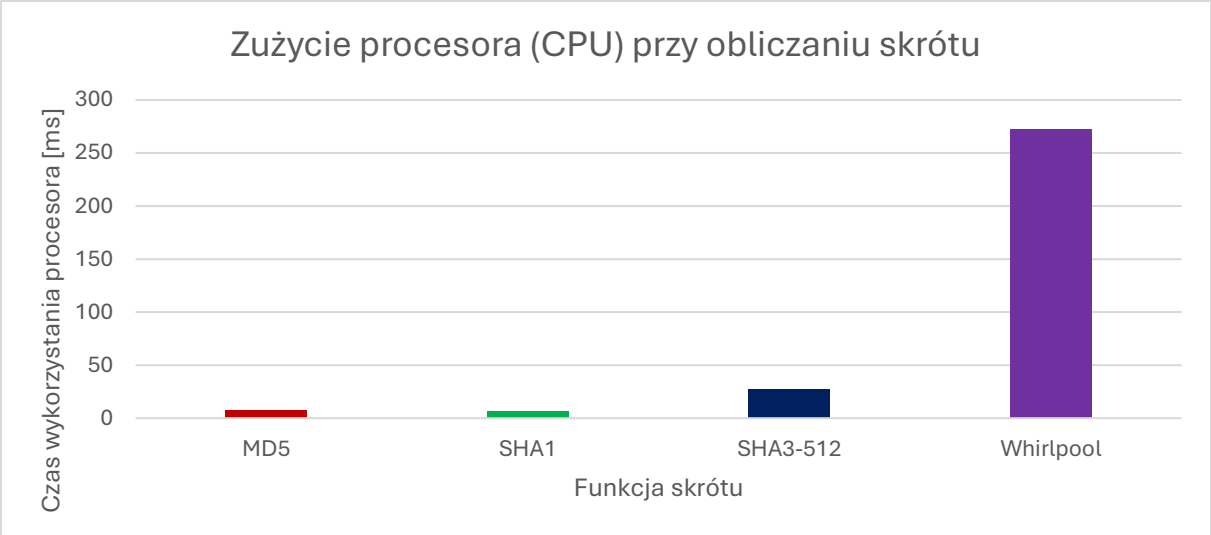
Rys. 22. Przepustowość

Przepustowość obliczono jako stosunek rozmiaru danych do czasu ich przetworzenia. Najwyższe wartości osiągnęły:

- **SHA-1 – 640 MB/s**,
- **MD5 – 570 MB/s**.
  SHA3-512 uzyskał **149 MB/s**, natomiast Whirlpool – **15 MB/s**. Różnice te potwierdzają, że złożoność algorytmu znacząco wpływa na szybkość działania, nawet przy prostych danych.

Tab. 22. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	null4216KB.txt	4,13	570,36
SHA1	null4216KB.txt	4,13	639,78
SHA3-512	null4216KB.txt	4,13	149,06
Whirlpool	null4216KB.txt	4,13	15,18



Rys. 23. Zużycie CPU

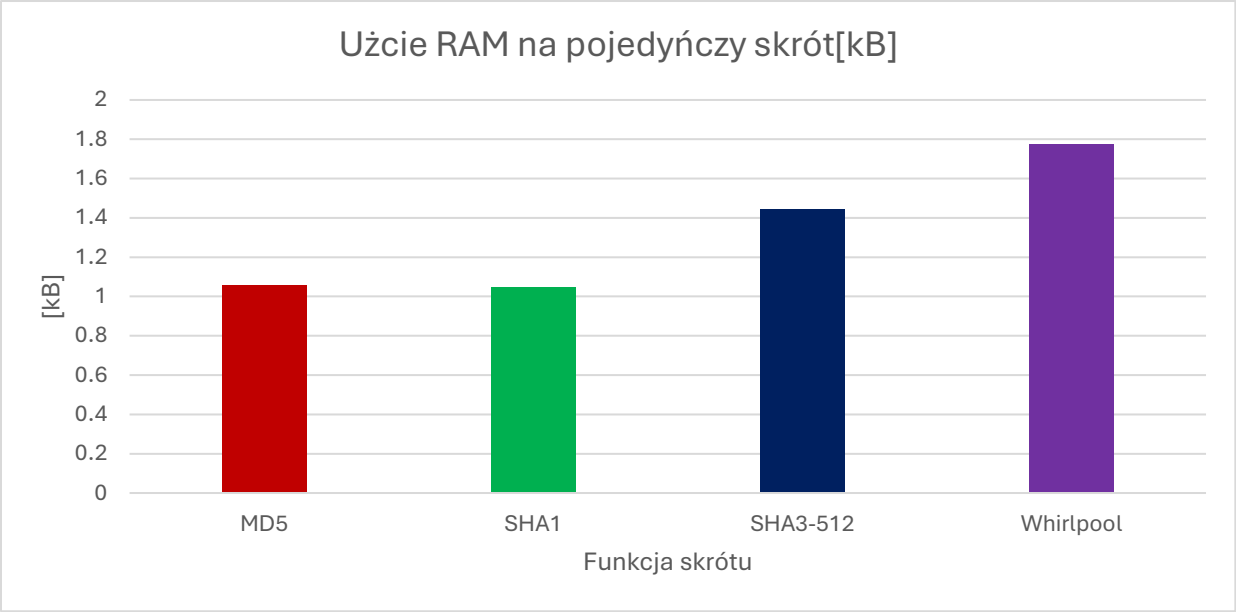
Na wykresie „Zużycie procesora" ponownie widzimy ogromną różnicę:

- **SHA-1**: 0,67 s,
- **MD5**: 0,72 s,
- **SHA3-512**: 2,77 s,
- **Whirlpool**: aż **27,2 s** czasu CPU.

Tab. 23. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	null4216KB.txt	4,13	7.19	7,19
SHA1	null4216KB.txt	4,13	6.72	6,72
SHA3-512	null4216KB.txt	4,13	27.66	27,66
Whirlpool	null4216KB.txt	4,13	272.34	272,34

W zakresie pamięci RAM, funkcje różniły się umiarkowanie:



Rys. 24. Zużycie RAM

- **SHA-1** i **MD5** – po około **1,05 kB**,
- **SHA3-512** – **1,44 kB**,
- **Whirlpool** – **1,77 kB**.

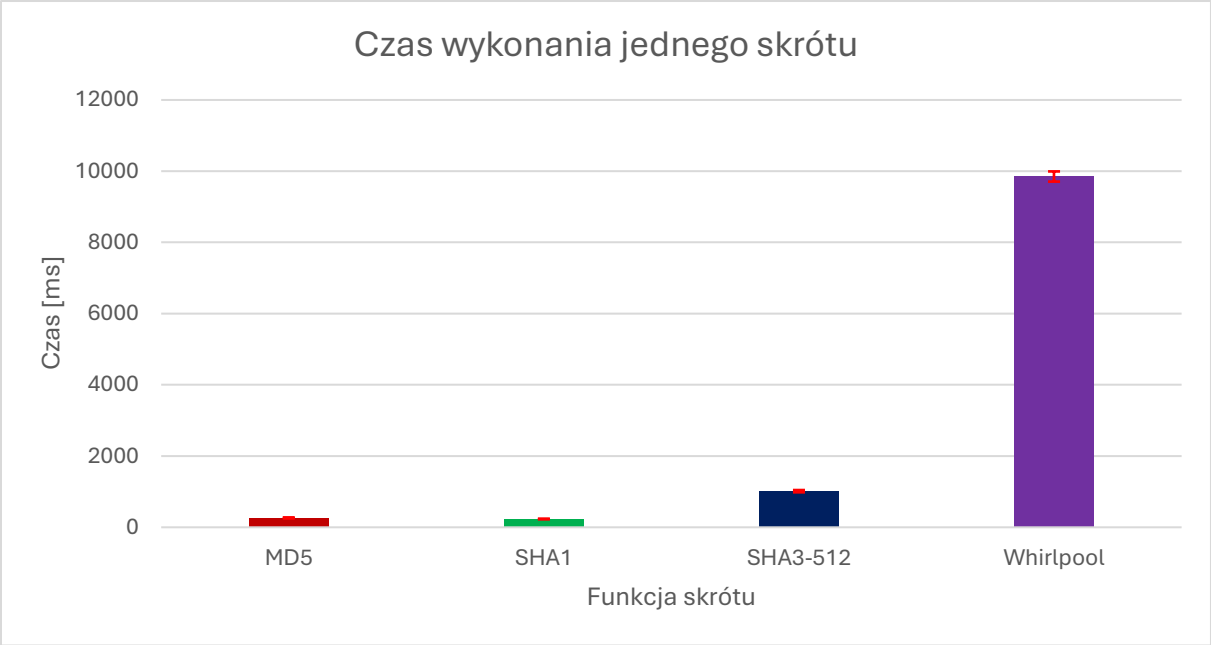
Wartości te pozostają stabilne w stosunku do wcześniejszych testów, potwierdzając charakterystyczne cechy każdej funkcji.

Tab. 24. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (MB)	Zużycie Pamięci (kB)
MD5	null4216KB.txt	4,13	0,103	105,472
SHA1	null4216KB.txt	4,13	0,102	104,448
SHA3-512	null4216KB.txt	4,13	0,141	144,384
Whirlpool	null4216KB.txt	4,13	0,173	177,152

4.7. Plik tekstowy wypełniony 0x00 149MB

W tym teście analizowano wydajność funkcji skrótu podczas przetwarzania bardzo dużego pliku tekstowego o rozmiarze **149 MB**, wypełnionego jednolitym bajtem 0x00. Tego typu dane wykorzystywane są często w testach syntetycznych, a także mogą występować w danych systemowych lub rezerwowanych strukturach plików.



Rys. 25. Czas wykonania

Jak pokazuje wykres „Czas wykonania jednego skrótu”, wyniki potwierdzają wcześniejsze obserwacje:

- **SHA-1**: 232,99 ms,
- **MD5**: 263,01 ms,
- **SHA3-512**: 1014,29 ms,
- **Whirlpool**: aż **9850,27 ms** (~10 s).

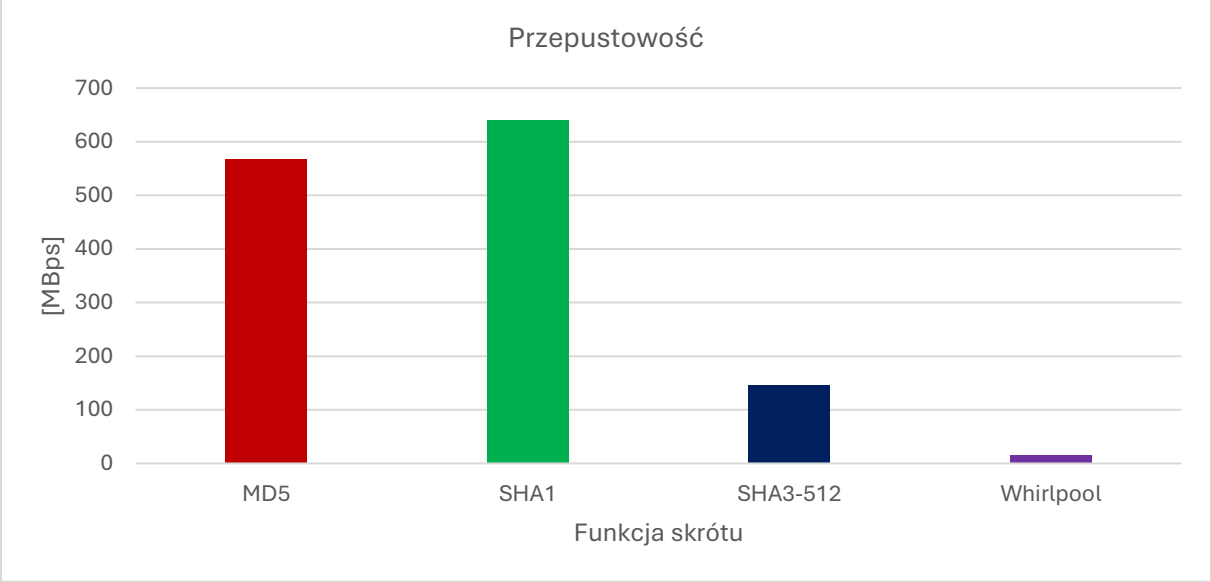
Stabilność funkcji przedstawia się następująco (odchylenie standardowe):

- **SHA-1**: 2,24 ms – najstabilniejszy wynik,
- **MD5**: 14,41 ms,
- **SHA3-512**: 31,34 ms,
- **Whirlpool**: 143,17 ms – duża zmienność czasowa, potwierdzona też przez wysokie percentyle (P99 = 10289 ms).

Tab. 25. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	null152MB.txt	149,09	263,01	14,405	258,638	276,046	344,737
SHA1	null152MB.txt	149,09	232,999	2,236	232,336	237,716	241,761

SHA 3-512	null152MB.txt	149,09	1014,286	31,344	1002,991	1062,286	1158,976
Whirlpool	null152MB.txt	149,09	9850,271	143,166	9810,332	10100,54	10289,26



Rys. 26. Przepustowość

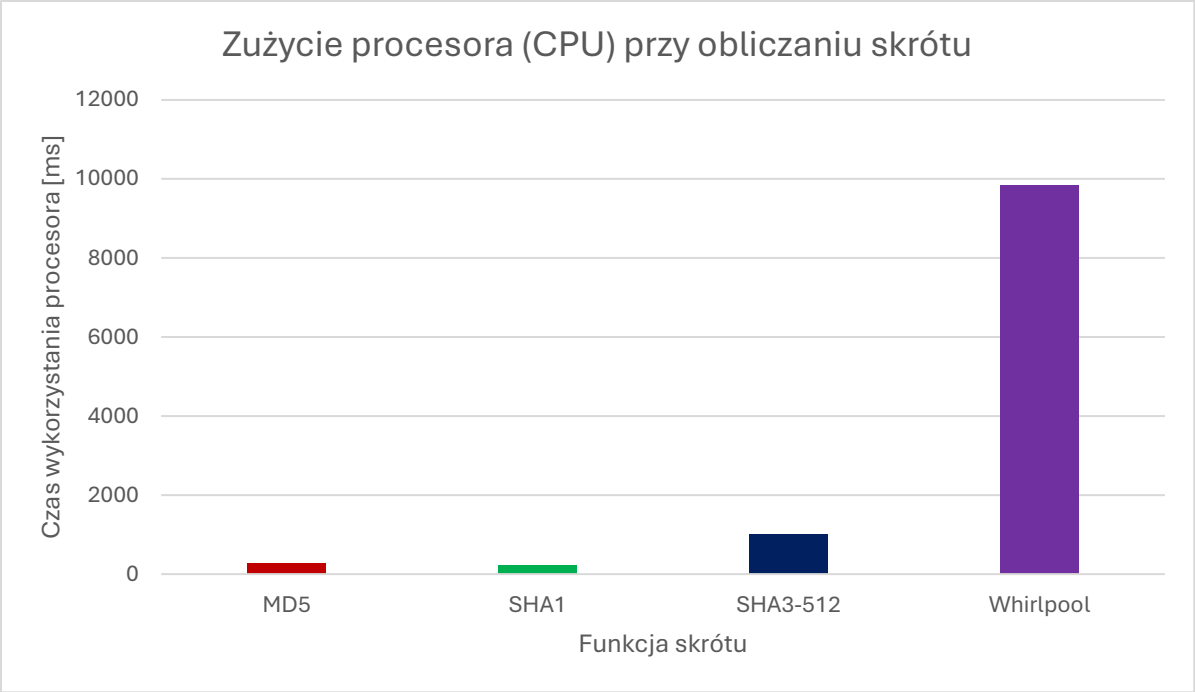
Z wykresu „Przepustowość" wynika, że:

- **SHA-1** osiągnął **640 MB/s**,
- **MD5**: **567 MB/s**,
- **SHA3-512**: **147 MB/s**,
- **Whirlpool**: tylko **15 MB/s**.

Funkcja Whirlpool ponownie wypada znacznie gorzej od pozostałych w kontekście przetwarzania dużych porcji danych.

Tab. 26. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	null152MB.txt	149,09	566,86
SHA1	null152MB.txt	149,09	639,87
SHA3-512	null152MB.txt	149,09	146,99
Whirlpool	null152MB.txt	149,09	15,14



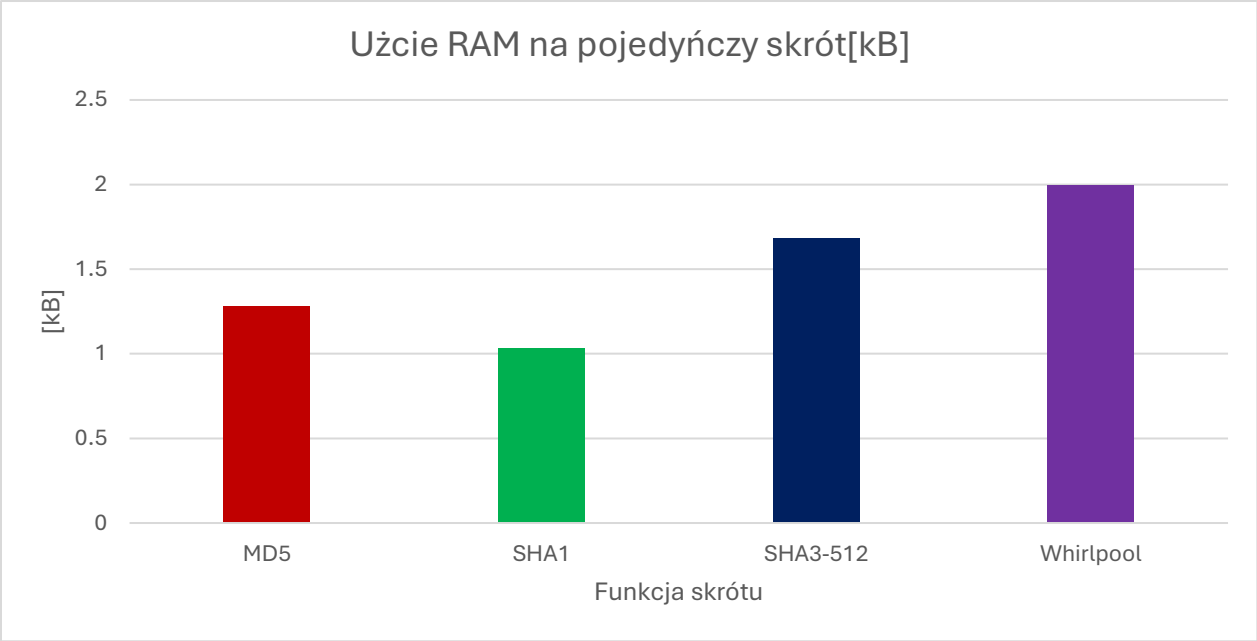
Rys. 27. Zużycie CPU

Czasy użycia procesora dla każdej funkcji przedstawiają się następująco:

- **SHA-1**: 23,3 s,
- **MD5**: 26,2 s,
- **SHA3-512**: 1013,1 s,
- **Whirlpool**: 9838,6 s (!).

Tab. 27. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	null152MB.txt	149,09	261.88	261,88
SHA1	null152MB.txt	149,09	233.28	233,28
SHA3-512	null152MB.txt	149,09	1013.12	1013,12
Whirlpool	null152MB.txt	149,09	9838.59	9838,59



Rys. 28. Zużycie RAM

- **SHA-1**: 1,03 kB,
- **MD5**: 1,28 kB,
- **SHA3-512**: 1,68 kB,
- **Whirlpool**: 2,00 kB.

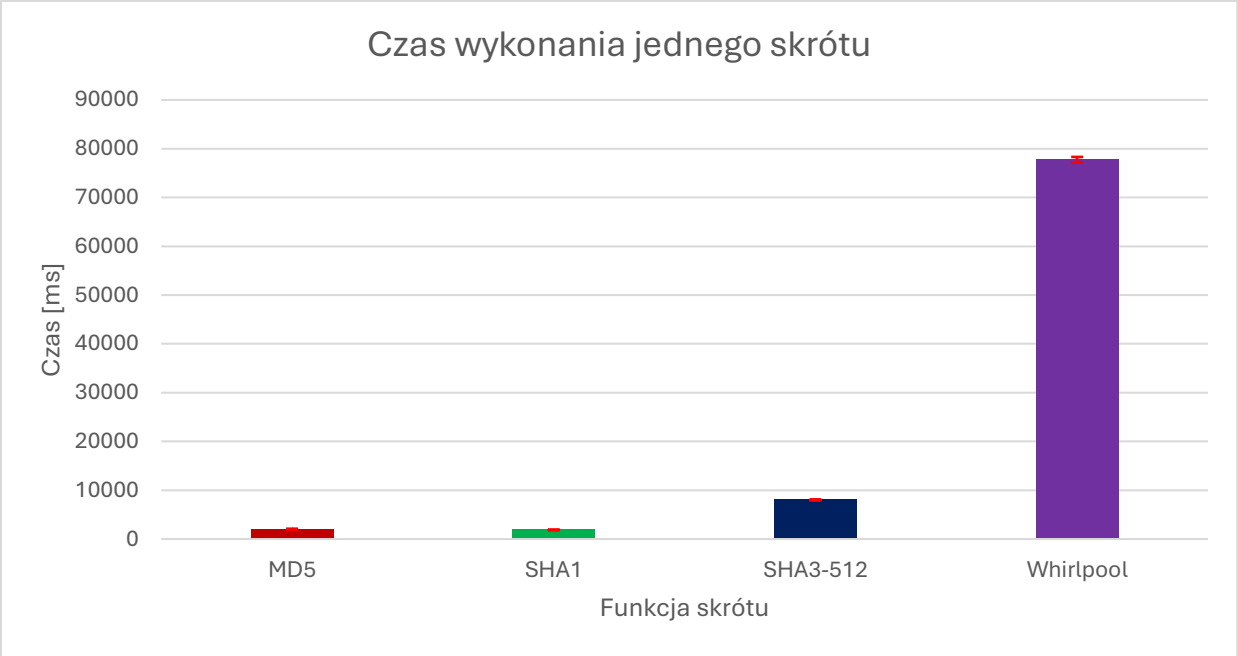
Pamięć operacyjna różni się niewiele, jednak Whirlpool wykorzystuje jej najwięcej – zarówno względnie, jak i absolutnie.

Tab. 28. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	null152MB.txt	149,09	128
SHA1	null152MB.txt	149,09	103,424
SHA3-512	null152MB.txt	149,09	167,936
Whirlpool	null152MB.txt	149,09	199,68

4.8. Plik tekstowy wypełniony 0x00 1.2GB

W ostatnim scenariuszu testowym przetworzono syntetyczny plik o rozmiarze **1179 MB (1,2 GB)**, w całości wypełniony bajtami 0x00. Dane tego typu odwzorowują skrajny przypadek dużych, jednorodnych strumieni informacji, które mogą występować w systemach przechowujących dane tymczasowe, archiwa lub pliki wymiany.



Rys. 29. Czas wykonania

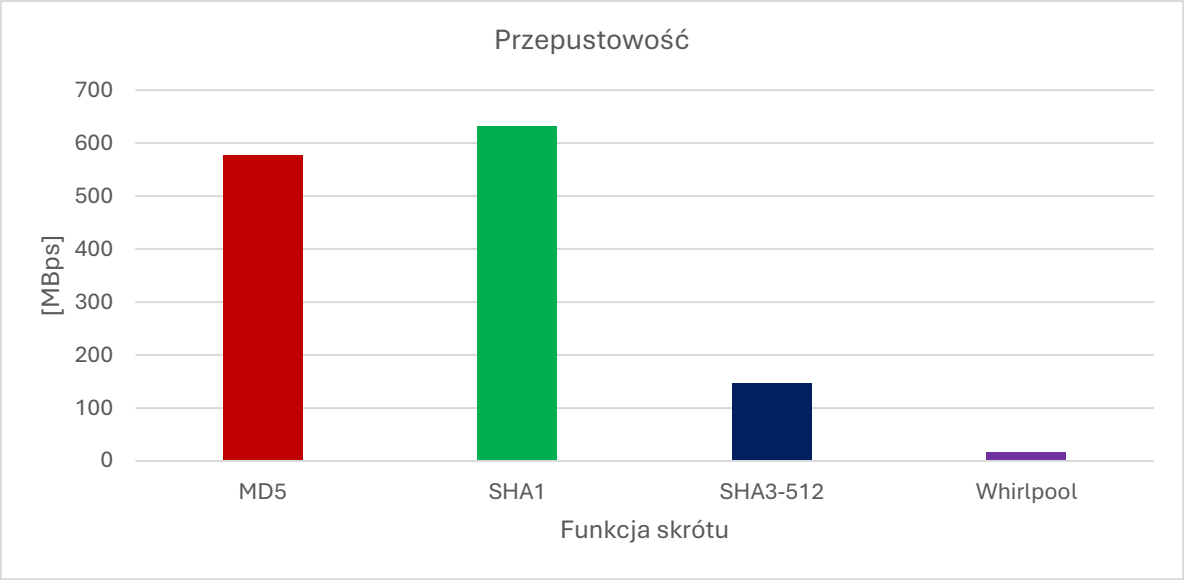
Zgodnie z wykresem „Czas wykonania jednego skrótu", funkcje SHA-1 i MD5 zachowały wysoką wydajność:

- **SHA-1**: 1865,77 ms,
- **MD5**: 2047,52 ms,
- **SHA3-512**: 8010,62 ms,
- **Whirlpool**: aż **77 755,33 ms** (~78 s).

Najbardziej stabilna okazała się SHA-1 – jej odchylenie standardowe wynosiło zaledwie 65,97 ms, co przekłada się na przewidywalność działania przy dużych danych.
Dla porównania, Whirlpool miał ogromną zmienność – aż **566,67 ms**.

Tab. 29. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	null1_2GB.txt	1179,38	2047,519	8,019	2044,49	2064,705	2074,715
SHA1	null1_2GB.txt	1179,38	1865,769	65,969	1841,092	1947,979	2095,951
SHA3-512	null1_2GB.txt	1179,38	8010,619	114,705	7973,618	8231,216	8453,06
Whirlpool	null1_2GB.txt	1179,38	77753	566,669	77625,33	78360,01	80511,84



Rys. 30. Przepustowość

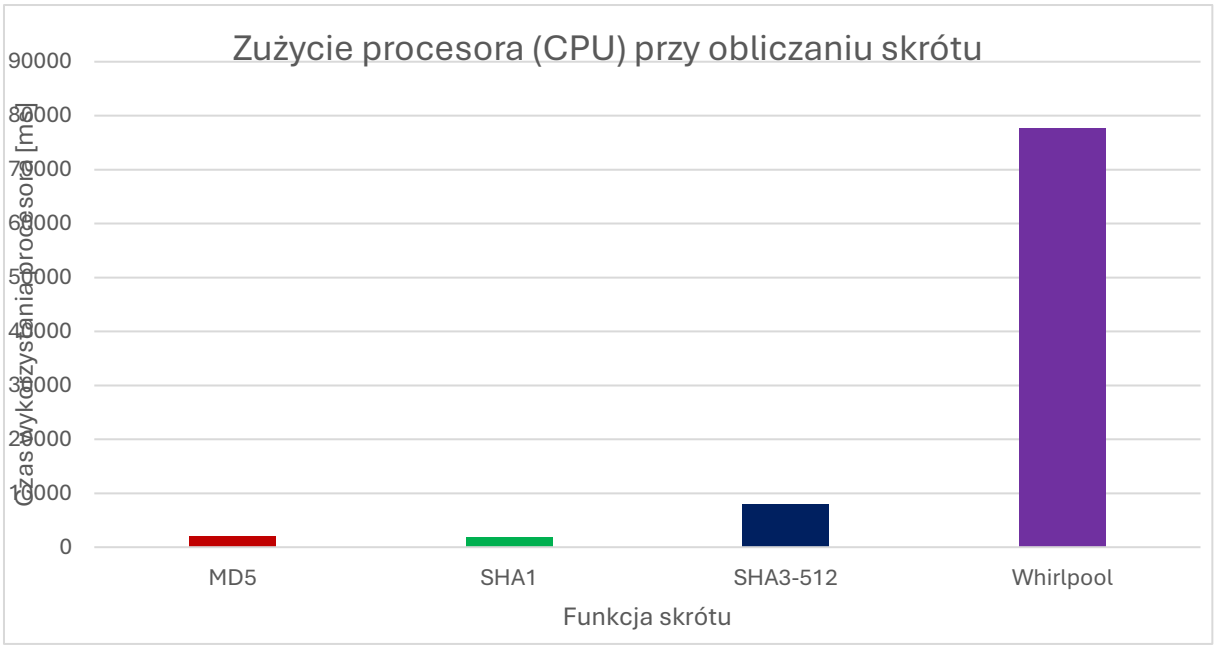
Wyniki przepustowości potwierdzają wyraźną przewagę SHA-1 i MD5 nad pozostałymi:

- **SHA-1**: 632 MB/s,
- **MD5**: 576 MB/s,
- **SHA3-512**: 147 MB/s,
- **Whirlpool**: tylko **15 MB/s**.

Wydajność SHA-1 w tym przypadku była najwyższa i najbardziej stabilna, co jest istotne w kontekście dużych zbiorów danych.

Tab. 30. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	null1_2GB.txt	1179,38	576
SHA1	null1_2GB.txt	1179,38	632,11
SHA3-512	null1_2GB.txt	1179,38	147,23
Whirlpool	null1_2GB.txt	1179,38	15,17



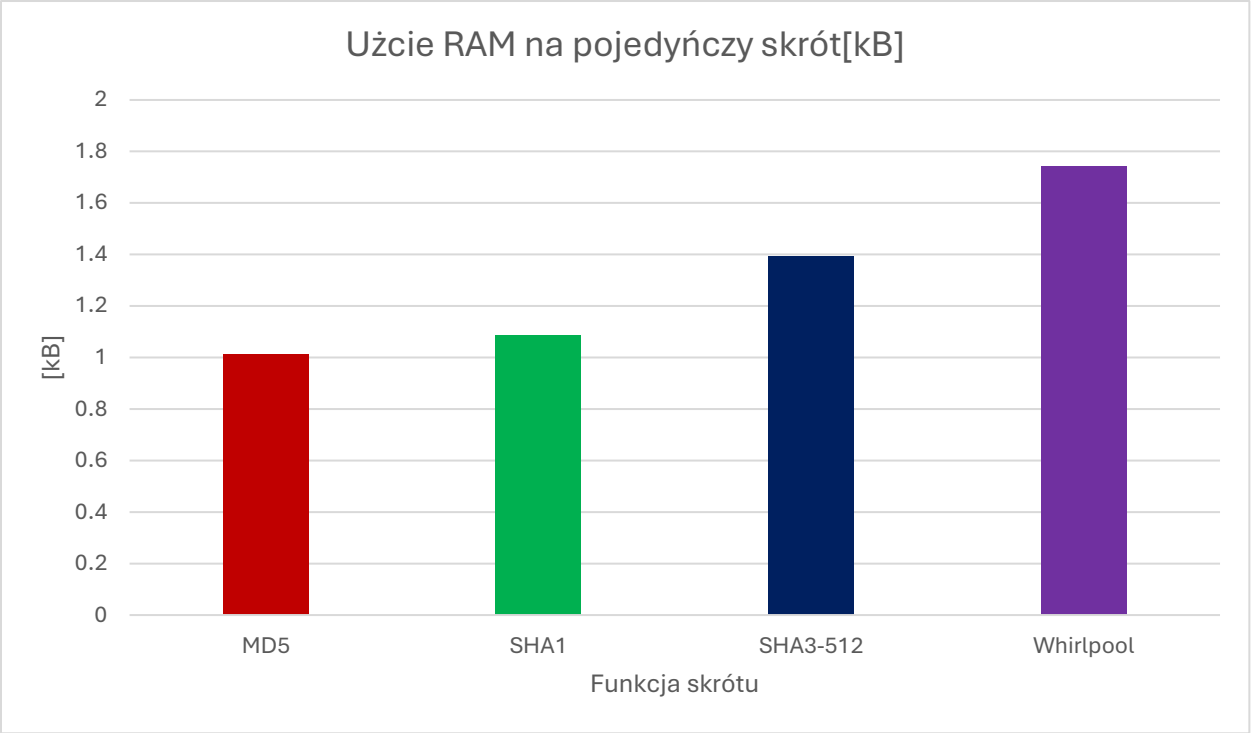
Rys. 31. Zużycie CPU

Z wykresu „Zużycie procesora” widać, że:

- **SHA-1** i **MD5** wymagają relatywnie mało zasobów CPU: 1861 s i 2046 s odpowiednio,
- **SHA3-512**: 8005 s,
- **Whirlpool**: aż **77 738 s**, co jest skrajnie nieefektywne.

Tab. 31. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	null1_2GB.txt	1179,38	2046.88	2046,88
SHA1	null1_2GB.txt	1179,38	1861.09	1861,09
SHA3-512	null1_2GB.txt	1179,38	8005.47	8005,47
Whirlpool	null1_2GB.txt	1179,38	77738.59	77738,59



Rys. 32. Zużycie RAM

- **SHA-1**: 1,08 kB,
- **MD5**: 1,01 kB,
- **SHA3-512**: 1,39 kB,
- **Whirlpool**: 1,74 kB.

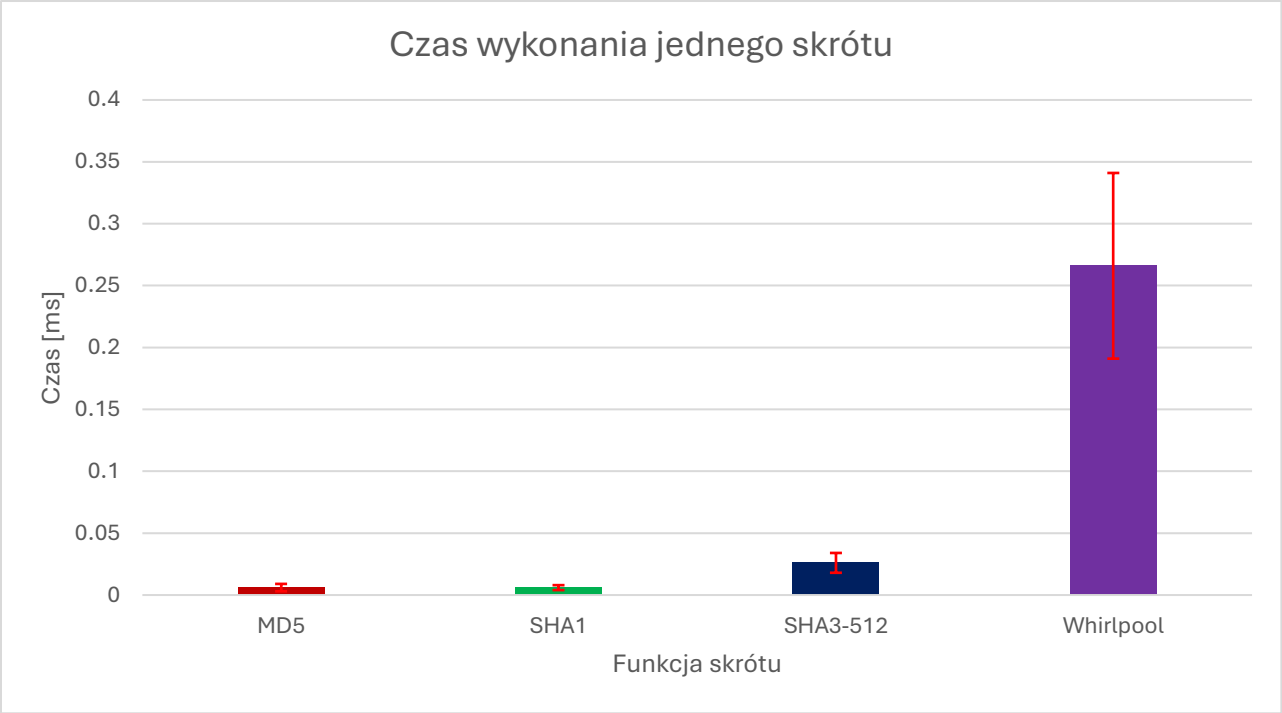
Zużycie pamięci utrzymuje się w podobnych proporcjach jak w poprzednich testach – największe dla Whirlpoola, najmniejsze dla MD5 i SHA-1.

Tab. 32. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	null1_2GB.txt	1179,38	101,376
SHA1	null1_2GB.txt	1179,38	108,544
SHA3-512	null1_2GB.txt	1179,38	139,264
Whirlpool	null1_2GB.txt	1179,38	174,08

4.9. Plik bin wypełniony losowymi bitami 3kB

Ostatni zestaw testów przeprowadzono na małym pliku binarnym o rozmiarze **3 kB**, wypełnionym losowymi bajtami. Takie dane reprezentują typowe ładunki binarne w komunikacji sieciowej, strukturach danych czy plikach tymczasowych, gdzie zróżnicowany rozkład bitów może wpłynąć na zachowanie funkcji skrótu.



Rys. 33. Czas wykonania

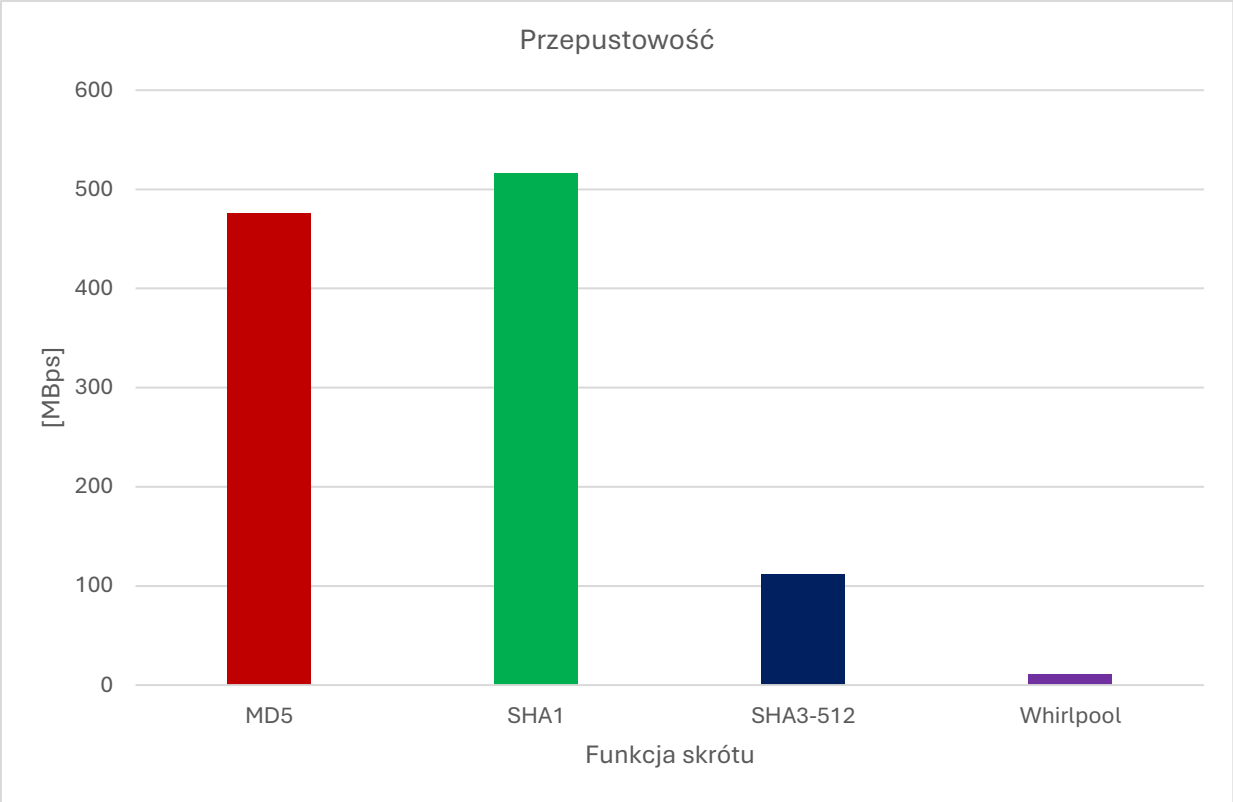
Wszystkie funkcje skrótu wykazały bardzo niską latencję:

- **MD5** i **SHA-1**: **0,006 ms**,
- **SHA3-512**: **0,026 ms**,
- **Whirlpool**: **0,266 ms**.

Odchylenia standardowe pozostawały niskie – dla MD5 i SHA-1 nie przekraczały **0,003 ms**, dla SHA3-512 – **0,008 ms**, natomiast Whirlpool odznaczał się największą zmiennością (0,075 ms). Wartości percentyli P99 wskazują na wysoką spójność MD5 i SHA-1 (0,024 ms i 0,026 ms) oraz wyraźną różnicę dla Whirlpoola (0,397 ms).

Tab. 33. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	Random 3KB.bin	0,006	0,003	0,006	0,006	0,024
SHA1	Random 3KB.bin	0,006	0,002	0,005	0,006	0,013
SHA3-512	Random 3KB.bin	0,026	0,008	0,022	0,039	0,047
Whirlpool	Random 3KB.bin	0,266	0,075	0,227	0,369	0,397



Rys. 34. Przepustowość

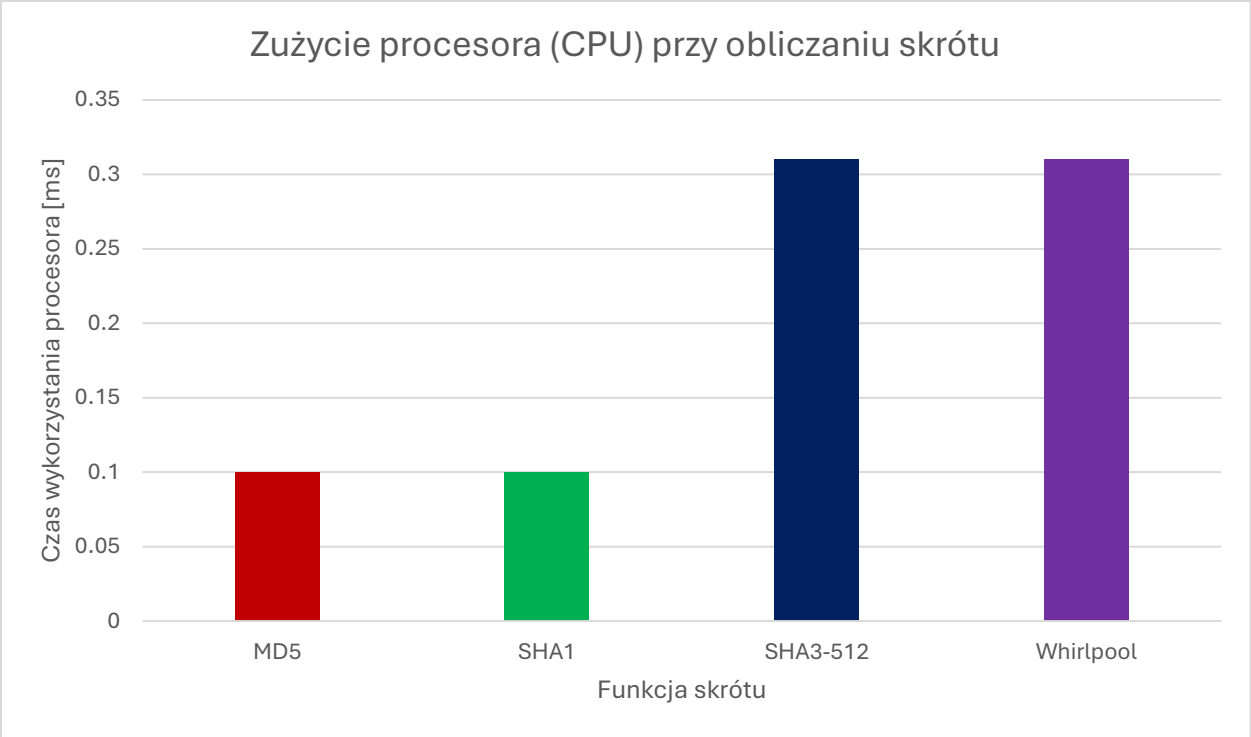
Dla tak małych plików przepustowość jest pośrednią miarą efektywności:

- **SHA-1 – 517 MB/s**,
- **MD5 – 476 MB/s**,
- **SHA3-512 – 112 MB/s**,
- **Whirlpool** – jedynie **11 MB/s**.

Mimo niewielkiego rozmiaru danych, różnice są zauważalne i wynikają bezpośrednio z konstrukcji algorytmu.

Tab. 34. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Przepustowość (MB/s)	Przepustowość (GB/s)
MD5	Random3KB.bin	475,98	0,4648
SHA1	Random3KB.bin	516,43	0,5043
SHA3-512	Random3KB.bin	112,09	0,1095
Whirlpool	Random3KB.bin	11,03	0,0108



Rys. 35. Zużycie CPU

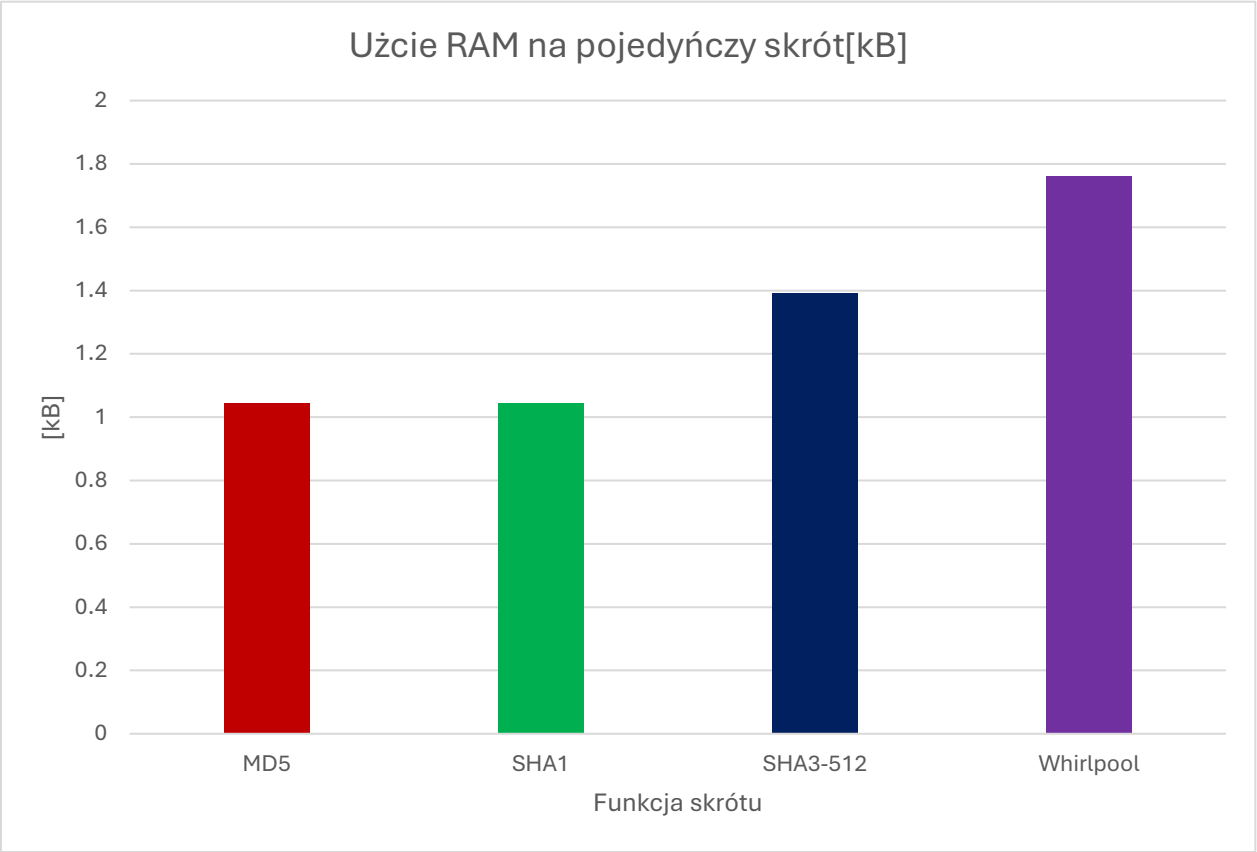
Z wykresu „Zużycie procesora" wynika, że:

- **MD5** i **SHA-1** potrzebowały tylko **0,10 ms** czasu CPU,
- **SHA3-512**: 0,31 ms,
- **Whirlpool**: 0,31 ms.

Tab. 35. Zestawienie wyników RAM

Algorytm	Typ Danych	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	Random3KB.bin	0.1	0,1
SHA1	Random3KB.bin	0.1	0,1
SHA3-512	Random3KB.bin	0.31	0,31
Whirlpool	Random3KB.bin	0.31	0,31

Wszystkie funkcje skrótu wykazały bardzo niskie wartości zużycia pamięci RAM:



Rys. 36. Zużycie RAM

- **MD5** i **SHA-1** – około **1,04 kB**,
- **SHA3-512** – **1,39 kB**,
- **Whirlpool** – **1,76 kB**.

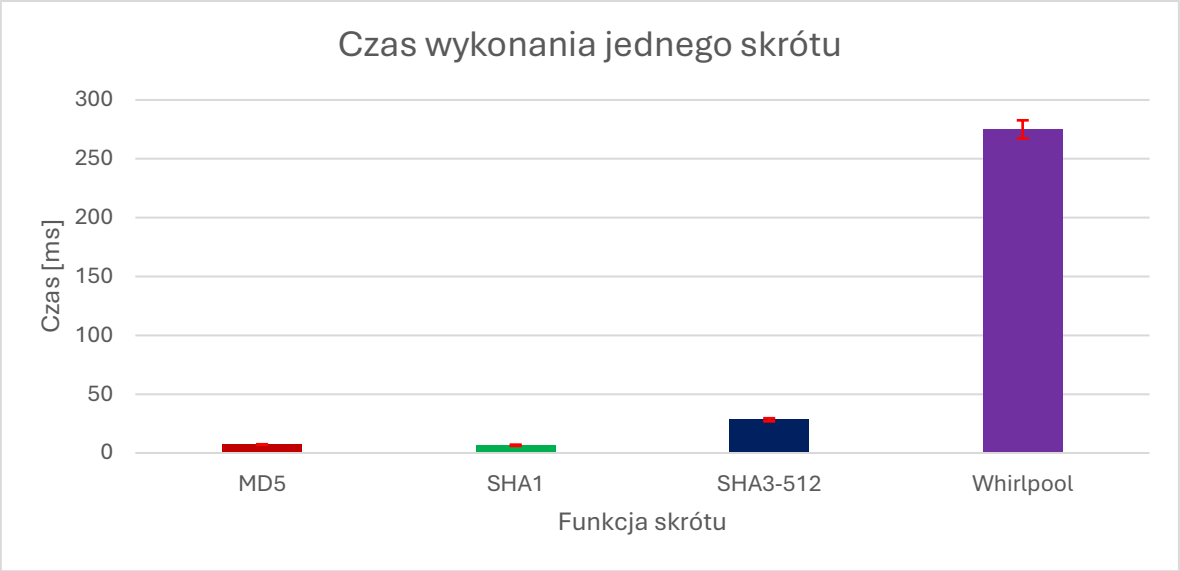
Dla bardzo małych danych wpływ na zasoby systemowe jest minimalny, ale zauważalne różnice wciąż mogą być istotne w środowiskach embedded lub systemach czasu rzeczywistego.

Tab. 36. Zestawienie wyników RAM

Algorytm	Typ Danych	Zużycie Pamięci (kB)
MD5	Random3KB.bin	104,448
SHA1	Random3KB.bin	104,448
SHA3-512	Random3KB.bin	139,264
Whirlpool	Random3KB.bin	176,128

4.10. Plik bin wypełniony losowymi bitami 4MB

W niniejszym teście wykorzystano binarny plik o wielkości **42,16 MB**, wypełniony losowymi danymi. Celem było sprawdzenie, jak funkcje skrótu radzą sobie z bardziej realistycznym, nieprzewidywalnym zestawem danych – typowym dla zaszyfrowanych plików, backupów lub zasobów multimedialnych.



Rys. 37. Czas wykonania

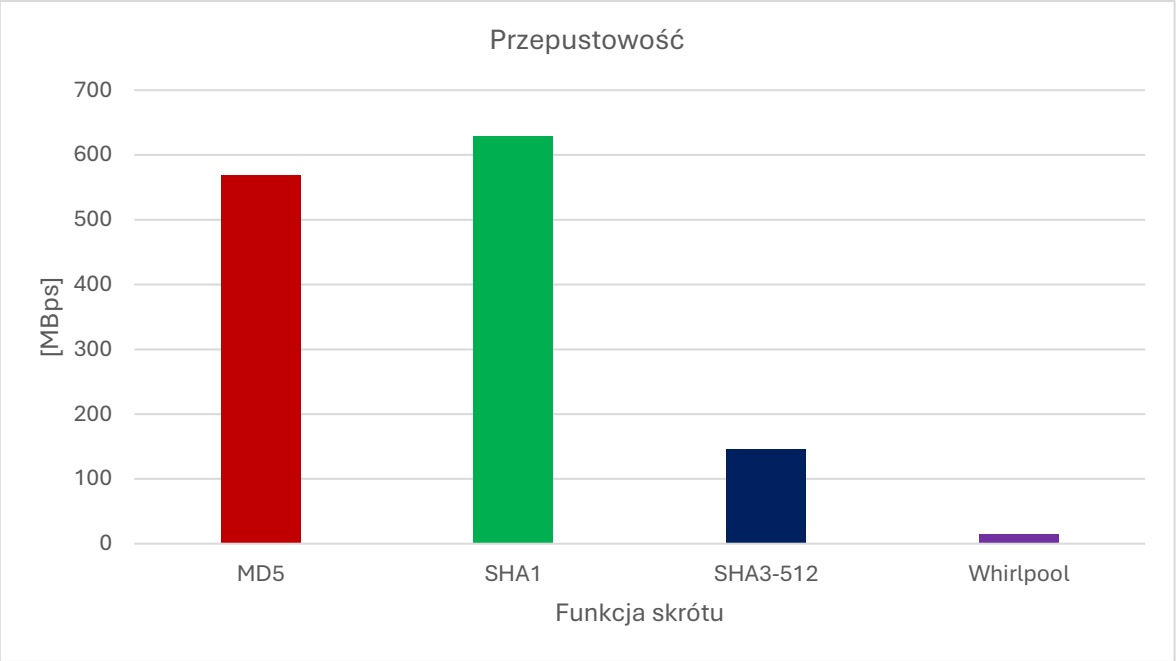
Tab. 37. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Czas wykonania	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	Random 4216KB.bin	4.216	7,233	0,173	7,226	7,61	7,817
SHA1	Random 4216KB.bin	4.216	6,553	0,545	6,449	6,735	9,048
SHA3-512	Random 4216KB.bin	4.216	28,218	1,314	27,771	29,714	35,417
Whirlpool	Random 4216KB.bin	4.216	274,995	7,706	271,414	290,668	306,118

Z wykresu „Czas wykonania jednego skrótu” wynika, że:

- **SHA-1** (6,55 ms) i **MD5** (7,23 ms) wykazały się najwyższą szybkością,
- **SHA3-512** potrzebował znacznie więcej czasu: 28,22 ms,
- **Whirlpool** – **274,99 ms**, czyli prawie dziesięciokrotnie więcej niż SHA3-512.

Odchylenia standardowe wskazują na stabilne działanie SHA-1 (0,55 ms) i MD5 (0,17 ms), natomiast Whirlpool charakteryzował się wysoką zmiennością (7,71 ms), co potwierdzają też jego wysokie wartości percentyli (P99 ≈ 306 ms).



Rys. 38. Przepustowość

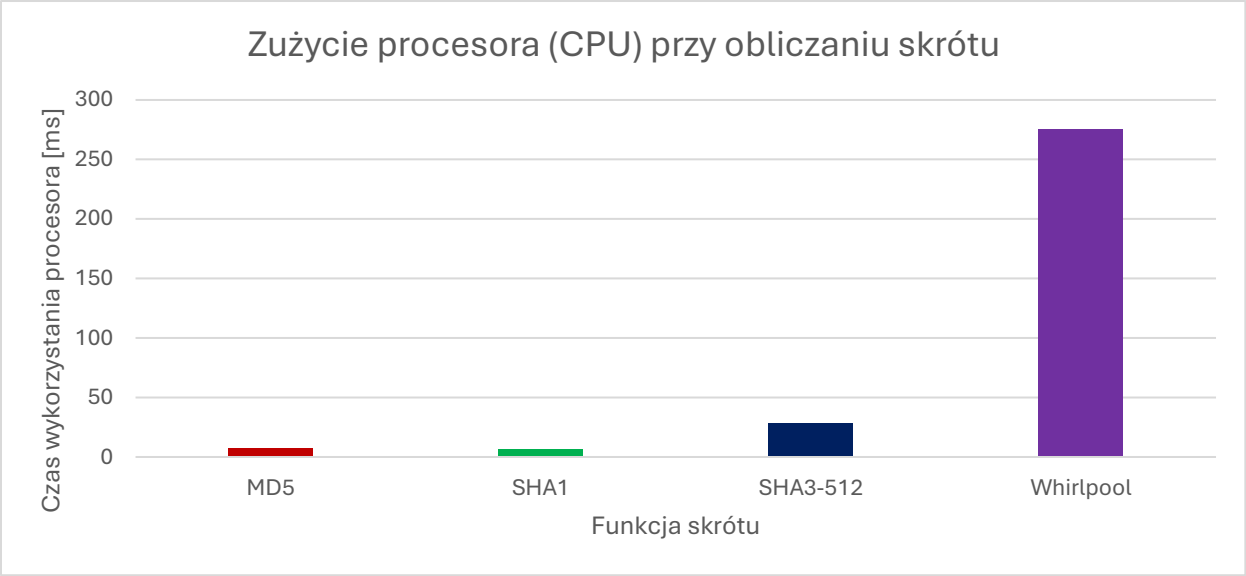
W kontekście efektywności przetwarzania danych:

- **SHA-1 – 629 MB/s**,
- **MD5 – 569 MB/s**,
- **SHA3-512 – 146 MB/s**,
- **Whirlpool – 15 MB/s**.

SHA-1 i MD5 uzyskały znacznie lepszą przepustowość, co czyni je odpowiednimi dla systemów przetwarzających większe ilości danych.

Tab. 38. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	Random4216KB.bin	4.216	569,23
SHA1	Random4216KB.bin	4.216	628,24
SHA3-512	Random4216KB.bin	4.216	145,91
Whirlpool	Random4216KB.bin	4.216	14,97



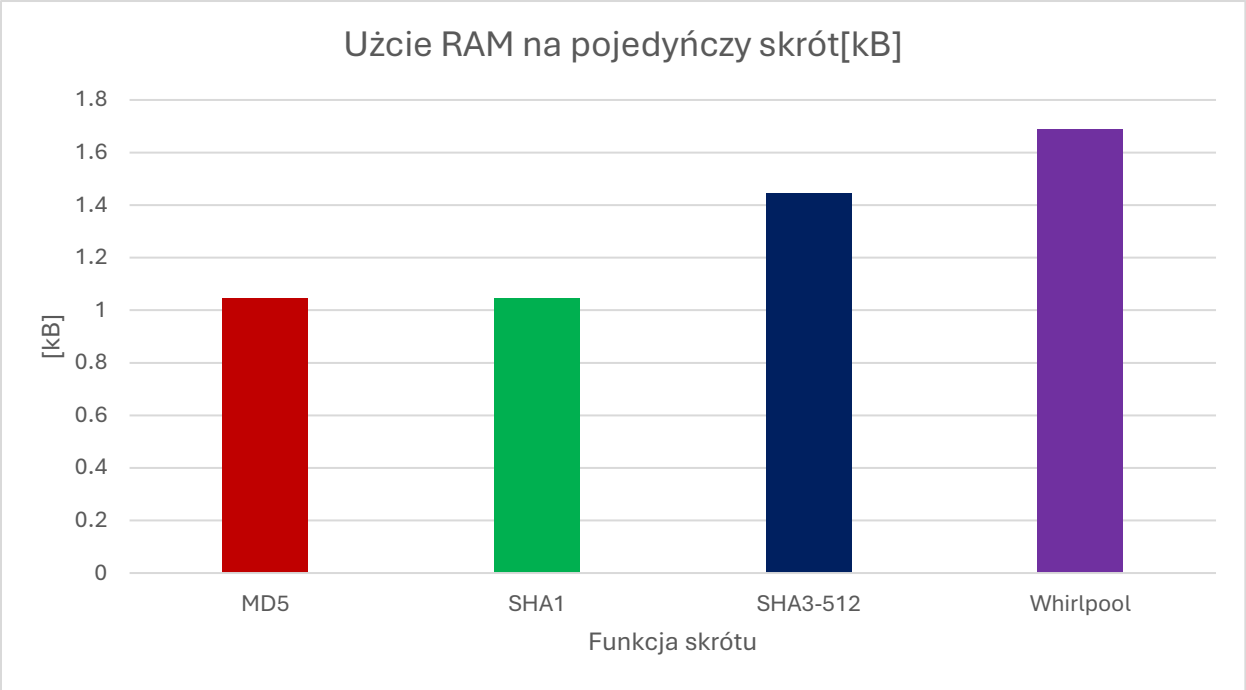
Rys. 39. Zużycie CPU

Na wykresie „Zużycie procesora" obserwujemy:

- **SHA-1** i **MD5** miały niskie zużycie CPU: 6,14 ms i 7,34 ms na iterację,
- **SHA3-512**: 28,28 ms,
- **Whirlpool**: aż **275 ms** – skrajnie nieefektywny pod względem CPU.

Tab. 39. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	Random4216KB.bin	4.216	7.34	7,34
SHA1	Random4216KB.bin	4.216	6.56	6,56
SHA3-512	Random4216KB.bin	4.216	28.28	28,28
Whirlpool	Random4216KB.bin	4.216	275	275



Rys. 40. Zużycie RAM

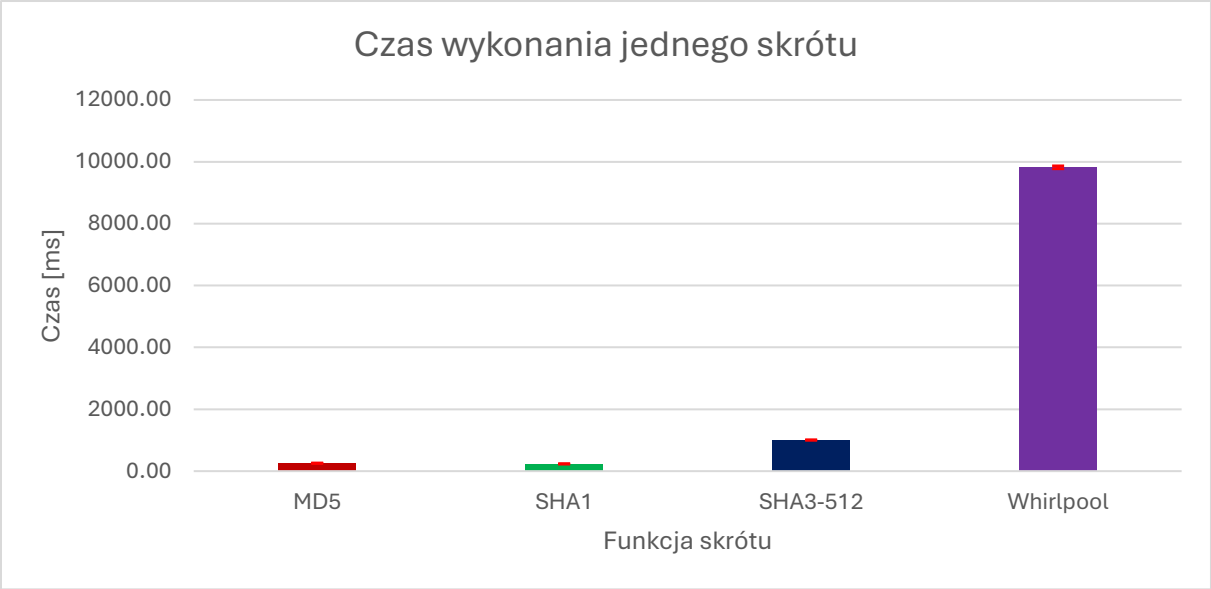
- **SHA-1** i **MD5** – po **1,04 kB**,
- **SHA3-512** – **1,44 kB**,
- **Whirlpool** – **1,69 kB**.

Tab. 40. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	Random4216KB.bin	4.216	104,448
SHA1	Random4216KB.bin	4.216	104,448
SHA3-512	Random4216KB.bin	4.216	144,384
Whirlpool	Random4216KB.bin	4.216	168,96

4.11. Plik bin wypełniony losowymi bitami 149MB

W ostatnim teście weryfikowano zachowanie funkcji skrótu przy przetwarzaniu dużego pliku binarnego o rozmiarze **149 MB**, wypełnionego losowymi bajtami. Taki scenariusz oddaje warunki rzeczywiste — dane nieskorelowane, występujące np. w plikach archiwalnych, obrazach systemowych, czy transmisjach zaszyfrowanych.



Rys. 41. Czas wykonania

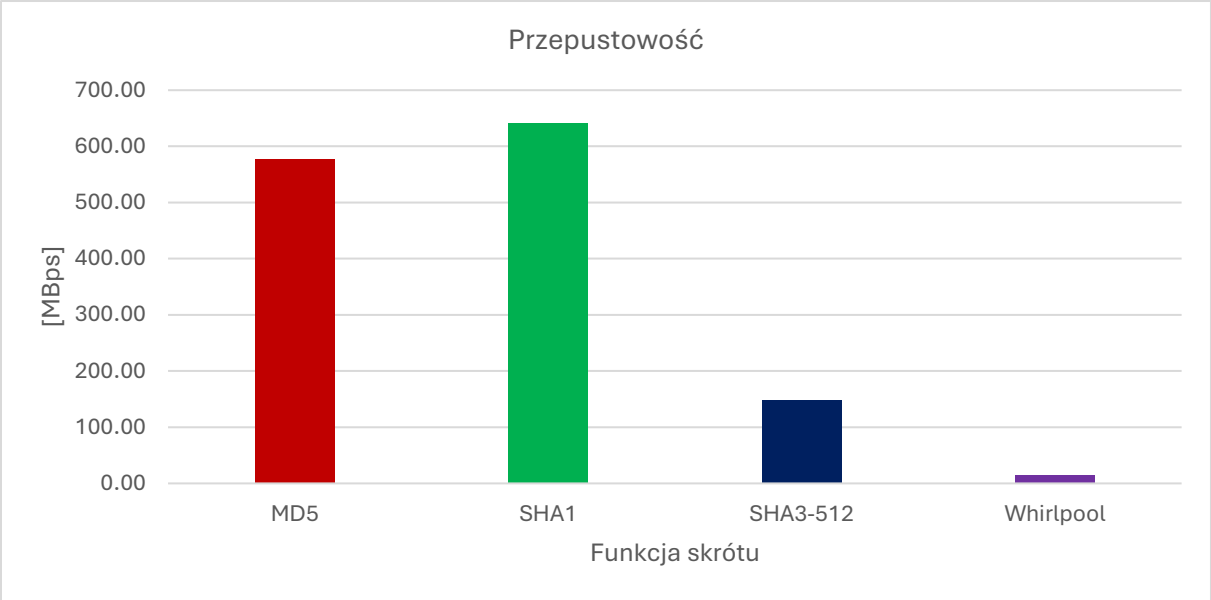
Zgodnie z wykresem „Czas wykonania jednego skrótu”:

- **SHA-1** osiągnął najniższy czas: **232,5 ms**,
- **MD5** był nieco wolniejszy: **258,4 ms**,
- **SHA3-512** potrzebował znacznie więcej – **1005,5 ms**,
- **Whirlpool** okazał się ponownie najwolniejszy: **9821 ms**.

Stabilność czasowa została potwierdzona przez niskie wartości odchylenia standardowego dla SHA-1 (1,46 ms) i MD5 (1,50 ms). W przypadku Whirlpoola wyniosło ono aż **48,58 ms**, co oznacza wyraźną zmienność w czasie przetwarzania.

Tab. 41. Zestawienie wyników latencji

Algorytm	Typ Danych	Rozmiar Danych (MB)	Odchylenie standardowe czasu (ms) wykonania	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	Random 152668KB.bin	149,09	1,50	257,84	262,26	263,48
SHA1	Random 152668KB.bin	149,09	1,46	232,09	235,06	239,24
SHA3-512	Random 152668KB.bin	149,09	8,14	1003,34	1017,98	1040,78
Whirlpool	Random 152668KB.bin	149,09	48,58	9804,89	9889,92	10017,78



Rys. 42. Przepustowość

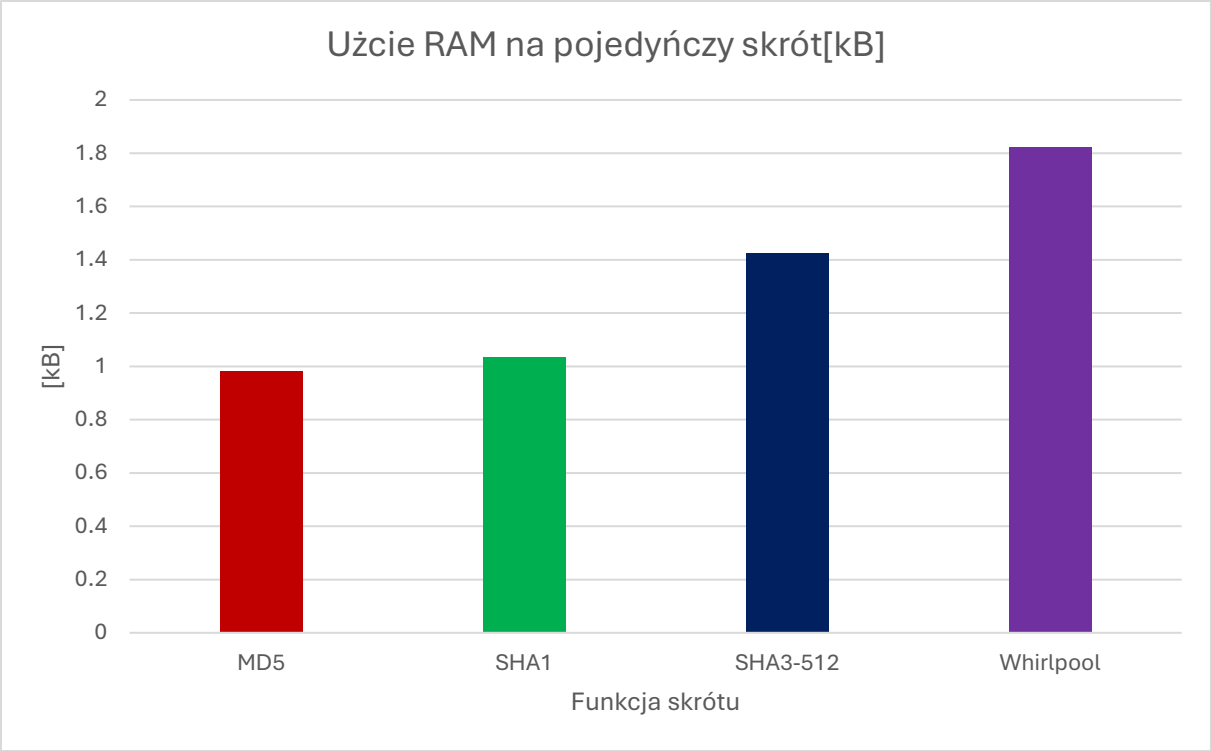
Najwyższe wartości przepustowości osiągnęły:

- **SHA-1**: **641 MB/s**,
- **MD5**: **577 MB/s**,
- **SHA3-512**: **148 MB/s**,
- **Whirlpool**: **15 MB/s**.

Przy losowych danych SHA-1 znów okazał się najwydajniejszym algorytmem, zarówno czasowo, jak i w kontekście ilości przetworzonych danych.

Tab. 42. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	Random152668KB.bin	149,09	577,09
SHA1	Random152668KB.bin	149,09	641,16
SHA3-512	Random152668KB.bin	149,09	148,26
Whirlpool	Random152668KB.bin	149,09	15,18



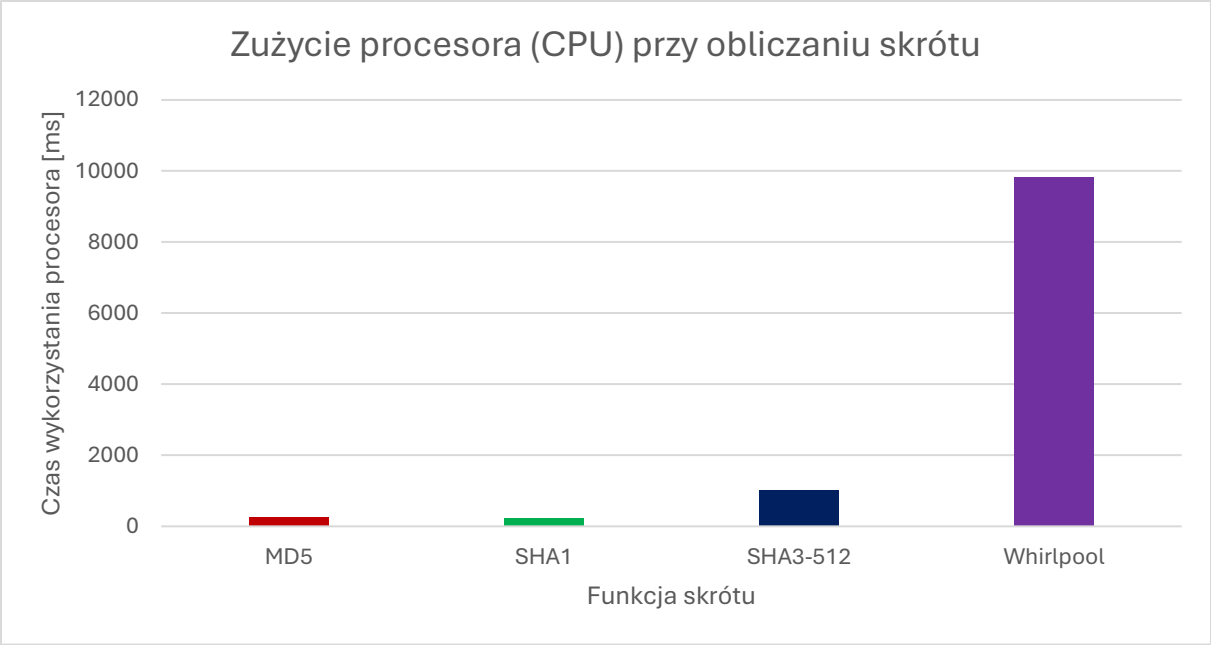
Rys. 43. Zużycie RAM

Z wykresu „Zużycie procesora" wynika, że:

- **SHA-1**: 232 ms CPU na iterację,
- **MD5**: 258 ms,
- **SHA3-512**: 1005 ms,
- **Whirlpool**: aż **9819 ms** (!).

Tab. 43. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	Random152668KB.bin	149,09	258.44	258,44
SHA1	Random152668KB.bin	149,09	232.5	232,5
SHA3-512	Random152668KB.bin	149,09	1005.47	1005,47
Whirlpool	Random152668KB.bin	149,09	9819.53	9819,53



Rys. 44. Zużycie CPU

- **SHA-1** i **MD5** – około **1,03–0,98 kB**,
- **SHA3-512** – **1,42 kB**,
- **Whirlpool** – **1,82 kB**.

Różnice są niewielkie, lecz istotne w środowiskach o ograniczonych zasobach (np. embedded, IoT).

Tab. 44. Zestawienie wyników RAM

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	Random152668KB.bin	149,09	98,304
SHA1	Random152668KB.bin	149,09	103,424
SHA3-512	Random152668KB.bin	149,09	142,336
Whirlpool	Random152668KB.bin	149,09	182,272

4.12. Plik bin wypełniony losowymi bitami 1.2GB

W ostatnim teście porównano wydajność funkcji skrótu na bardzo dużym pliku binarnym o rozmiarze **1179 MB**, zawierającym losowo rozmieszczone dane binarne. Jest to przykład typowego przypadku rzeczywistego — plik zaszyfrowany, skompresowany lub pochodzący z transmisji sieciowej.

Rys. 45. Czas wykonania

Z wykresu „Czas wykonania jednego skrótu" wynika, że:

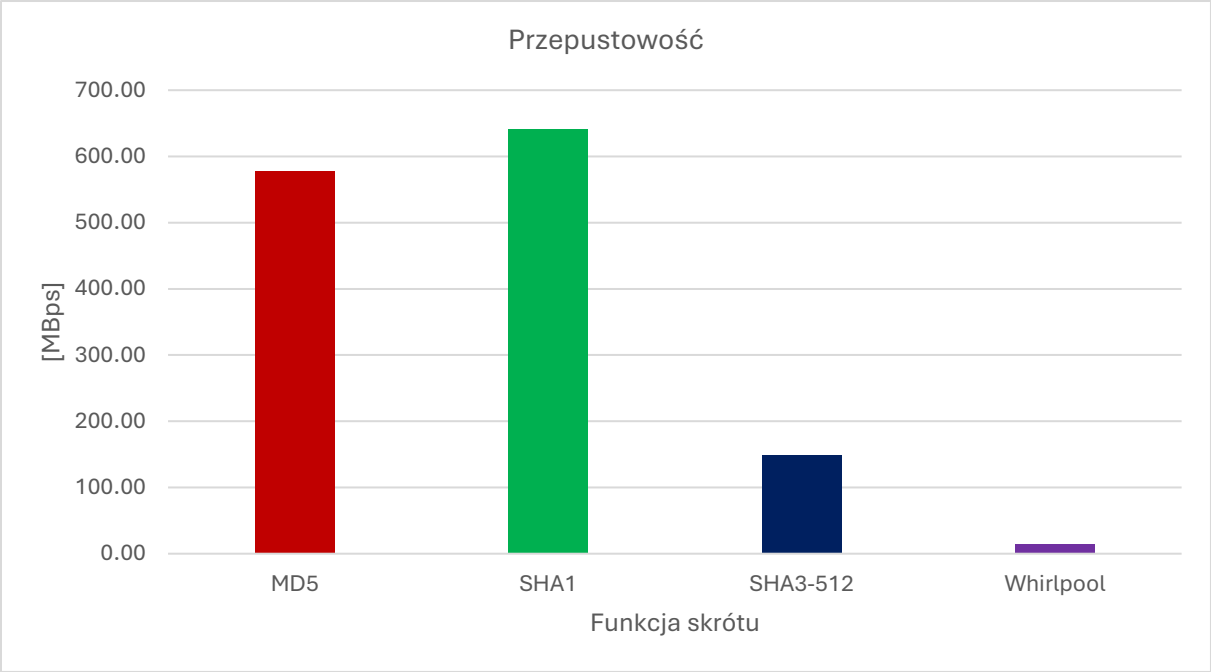
- **SHA-1** był najszybszy: **1840,55 ms**,
- **MD5** tuż za nim: **2043,32 ms**,
- **SHA3-512** potrzebował znacznie więcej: **7940,28 ms**,
- **Whirlpool**: **79897,35 ms** (~80 s).

Pod względem stabilności czasowej:

- **SHA-1** i **MD5** wykazały bardzo niskie odchylenia standardowe: odpowiednio **12,6 ms** i **6,2 ms**,
- **SHA3-512**: 33,5 ms,
- **Whirlpool**: 2500 ms — bardzo wysoka niestabilność.

Tab. 45. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Czas wykonania	Odchylenie standardowe	Opóźnienie P50 (ms)	Opóźnienie P95 (ms)	Opóźnienie P99 (ms)
MD5	Random1_2.bin	1179,03	2043,324	6,19	2041,38	2058,94	2063,45
SHA1	Random1_2.bin	1179,03	1840,55	12,59	1836,22	1855,96	1902,90
SHA3-512	Random1_2.bin	1179,03	7940,283	33,54	7925,88	8010,59	8035,33
Whirlpool	Random1_2.bin	1179,03	79897,35	2500,43	78981,98	84332,38	85253,45



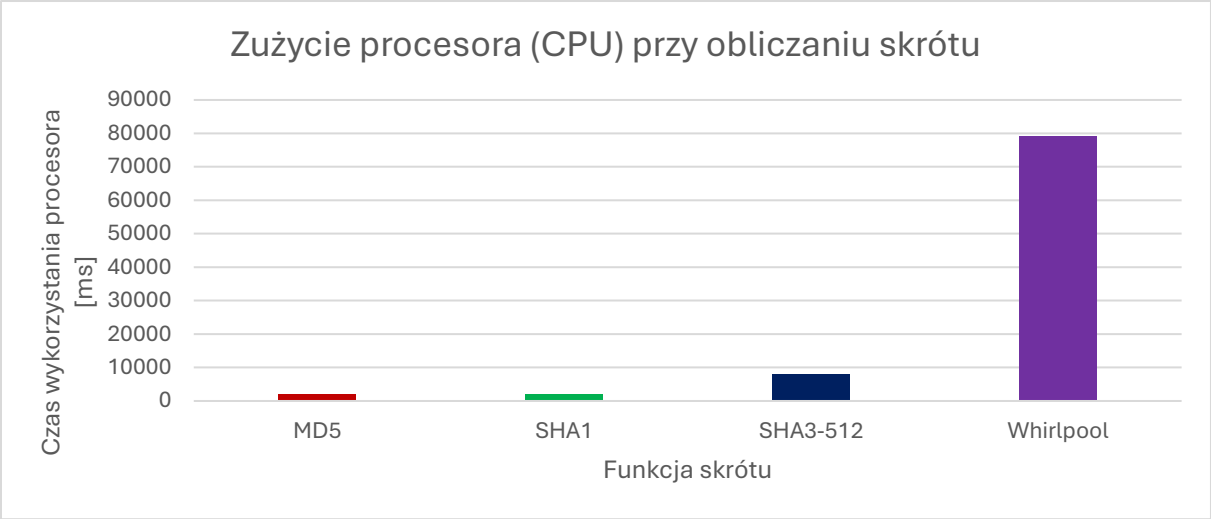
Rys. 46. Przepustowość

W zakresie wydajności przetwarzania:

- **SHA-1** osiągnął najwyższą przepustowość: **640 MB/s**,
- **MD5**: **577 MB/s**,
- **SHA3-512**: **148 MB/s**,
- **Whirlpool**: jedynie **15 MB/s**.

Tab. 46. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Przepustowość (MB/s)
MD5	Random1_207_328KB.bin	1179,03	577,02
SHA1	Random1_207_328KB.bin	1179,03	640,59
SHA3-512	Random1_207_328KB.bin	1179,03	148,49
Whirlpool	Random1_207_328KB.bin	1179,03	14,76



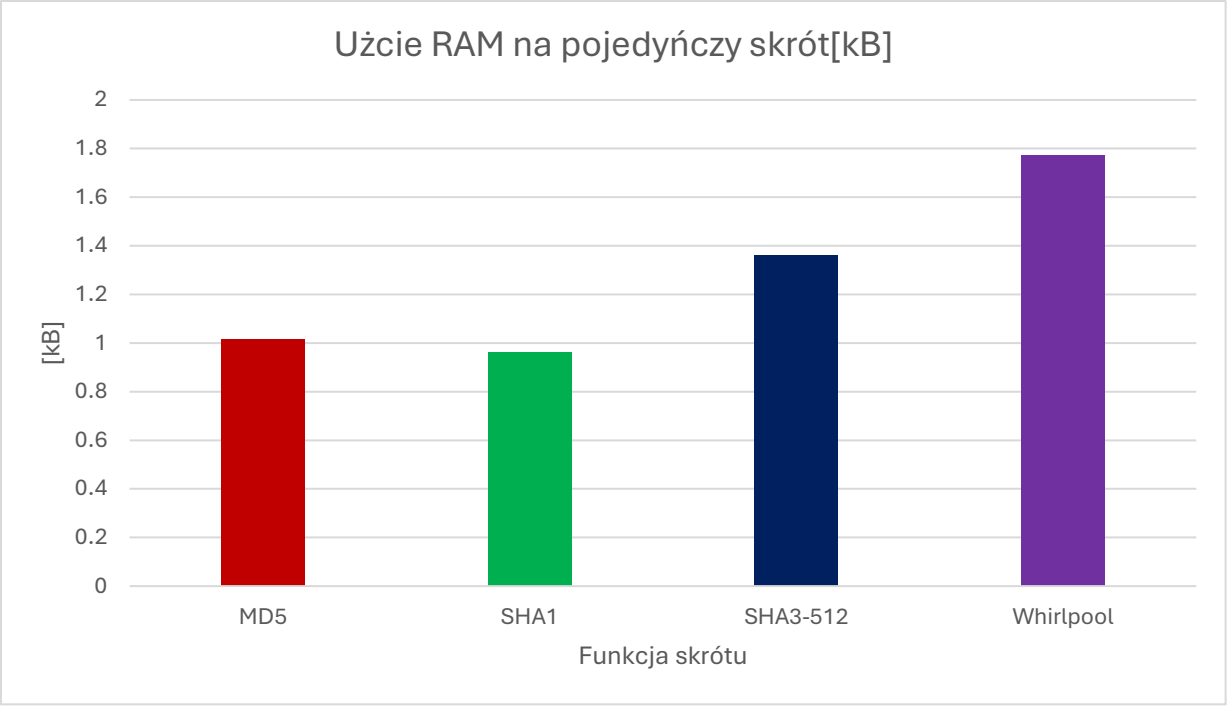
Rys. 47. Zużycie CPU

Z wykresu „Zużycie procesora”:

- **SHA-1**: **1840 ms**,
- **MD5**: **2043 ms**,
- **SHA3-512**: **7938 ms**,
- **Whirlpool**: **79054 ms** – wartość wykluczająca ten algorytm z użycia.

Tab. 47. Zestawienie wyników CPU

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie CPU (ms)	CPU na iteracje [ms]
MD5	Random1_207_328KB.bin	1179,03	2043.12	2043,12
SHA1	Random1_207_328KB.bin	1179,03	1840.31	1840,31
SHA3-512	Random1_207_328KB.bin	1179,03	7938.91	7938,91
Whirlpool	Random1_207_328KB.bin	1179,03	79054.53	79054,53



Rys. 48. Zużycie RAM

- **SHA-1**: **0,96 kB**,
- **MD5**: **1,01 kB**,
- **SHA3-512**: **1,36 kB**,
- **Whirlpool**: **1,77 kB**.

Tab. 48. Zestawienie wyników przepustowości

Algorytm	Typ Danych	Rozmiar Danych (MB)	Zużycie Pamięci (kB)
MD5	Random1_207_328KB.bin	1179,03	101,376
SHA1	Random1_207_328KB.bin	1179,03	96,256
SHA3-512	Random1_207_328KB.bin	1179,03	136,192
Whirlpool	Random1_207_328KB.bin	1179,03	177,152

4.13. Analiza stabilności i zmienności wyników

Jednym z kluczowych aspektów oceny funkcji skrótu – poza wydajnością – jest stabilność wyników, czyli powtarzalność czasu wykonania operacji przy wielokrotnym przetwarzaniu tych samych danych. W analizie wykorzystano odchylenie standardowe oraz percentyle (P50, P95, P99) jako metryki pozwalające ocenić rozrzut czasowy.

MD5 i SHA-1

Funkcje MD5 i SHA-1 konsekwentnie wykazywały bardzo niskie odchylenia standardowe – zazwyczaj poniżej 2 ms dla dużych plików, i ułamki milisekundy przy małych. Percentyle P95 i P99 różniły się od mediany zaledwie minimalnie, co świadczy o bardzo dużej powtarzalności. Te cechy czynią je stabilnymi i przewidywalnymi w środowiskach produkcyjnych o ograniczonej tolerancji na zmienność czasu wykonania.

SHA3-512

SHA3-512 również zachowywała dość stabilny profil, choć jej odchylenia standardowe były wyższe (ok. 30–100 ms przy dużych danych). Percentyle wskazują na lekkie przesunięcia w czasie wykonania (P99 nieznacznie odbiega od mediany), co wciąż mieści się w akceptowalnym zakresie. Algorytm można uznać za umiarkowanie stabilny, szczególnie biorąc pod uwagę jego większą złożoność.

Whirlpool

Whirlpool był zdecydowanie najbardziej niestabilną funkcją – jego odchylenia standardowe przekraczały 500 ms, a często wynosiły nawet kilka sekund (przy plikach 1–2 GB). Różnice pomiędzy wartościami P50 i P99 były bardzo duże, co oznacza wysoką zmienność czasu działania, nieprzewidywalną z punktu widzenia systemu. W środowiskach wymagających deterministycznej wydajności – funkcja ta nie powinna być stosowana.

4.14. Podsumowanie wyników – porównawcze zestawienie

Poniżej przedstawiono zbiorczą ocenę funkcji skrótu MD5, SHA-1, SHA3-512 i Whirlpool-512 na podstawie przeprowadzonych testów. Ocena obejmuje pięć kryteriów: **czas wykonania**, **przepustowość**, **zużycie CPU**, **zużycie RAM** oraz **stabilność wyników**.

Tab. 49. Podsumowane wyniki

Funkcja	Czas wykonania	Przepustowość	Zużycie CPU	Zużycie RAM	Stabilność
SHA-1	bardzo szybka	najwyższa (~640 MB/s)	niskie	najniższe (~1 kB)	bardzo wysoka
MD5	szybka	bardzo dobra (~570 MB/s)	niskie	niskie (~1 kB)	bardzo wysoka
SHA3-512	wolniejsza	średnia (~140 MB/s)	wyższe	umiarkowane (~1.4 kB)	dobra
Whirlpool	bardzo wolna	bardzo niska (~15 MB/s)	ekstremalne	najwyższe (~1.8–2 kB)	bardzo niska

5. Wnioski

5.1. Najbardziej optymalna funkcja skrótu w zależności od kryterium

Na podstawie wyników testów, można wyodrębnić optymalne funkcje skrótu w zależności od konkretnego kryterium:

Tab. 50. Zalety funkcji skrótu

Kryterium	Najlepsza funkcja	Uzasadnienie
Czas wykonania (latencja)	SHA-1	Najszybsza funkcja w większości scenariuszy testowych.
Przepustowość (MB/s)	SHA-1	Największa ilość danych przetwarzana w jednostce czasu.
Stabilność działania	SHA-1 / MD5	Niskie odchylenia standardowe i przewidywalne percentyle.
Zużycie CPU	SHA-1 / MD5	Niskie wartości czasu procesora nawet przy dużych plikach.
Zużycie RAM	SHA-1	Najniższe zużycie pamięci operacyjnej na iterację.
Bezpieczeństwo kryptograficzne	SHA3-512	Odporna na kolizje, nowoczesna konstrukcja (Keccak).

5.2. Możliwości zastosowania badanych funkcji w praktyce

Każda z badanych funkcji może znaleźć zastosowanie w innych obszarach — w zależności od priorytetów takich jak wydajność, bezpieczeństwo czy ograniczenia środowiskowe:

MD5

- Zastosowanie: sumy kontrolne plików, wykrywanie duplikatów, systemy backupu.
- Uwagi: pomimo wysokiej wydajności, niezalecana w kontekście bezpieczeństwa (kolizje).
- Środowiska: systemy lokalne, automatyczne przetwarzanie danych offline.

SHA-1

- Zastosowanie: kontrola integralności danych, wersjonowanie (Git), systemy operacyjne.
- Uwagi: nadal stosowana w wielu starszych systemach, ale powoli wycofywana z zastosowań kryptograficznych.
- Środowiska: systemy wbudowane, aplikacje desktopowe, narzędzia skryptowe.

SHA3-512

- Zastosowanie: kryptografia, podpisy cyfrowe, blockchain, certyfikacja danych.
- Uwagi: wysoka odporność na kolizje i nowoczesna konstrukcja (Keccak), kosztem niższej wydajności.
- Środowiska: aplikacje bezpieczeństwa, systemy bankowe, ochrona danych.

Whirlpool

- Zastosowanie: archiwizacja, systemy zgodne z europejskimi standardami bezpieczeństwa.
- Uwagi: wysokie zużycie CPU i RAM oraz niska stabilność sprawiają, że Whirlpool ma ograniczone zastosowanie praktyczne.
- Środowiska: niszowe zastosowania zgodne z wymaganiami specyficznych standardów.

Wnioski końcowe:
W analizie technicznej i wydajnościowej SHA-1 okazała się najbardziej uniwersalną i optymalną funkcją skrótu do zastosowań ogólnych, gdzie bezpieczeństwo nie jest głównym priorytetem.
W kontekście bezpieczeństwa SHA3-512 pozostaje najlepszym wyborem, mimo wyższych kosztów obliczeniowych.
MD5 można nadal stosować w aplikacjach niewymagających odporności kryptograficznej, natomiast Whirlpool – ze względu na jego nieefektywność – nie jest zalecany w zastosowaniach produkcyjnych.

6. Bibliografia

Porównanie szybkości działania wybranych funkcji skrótu i algorytmów szyfrowania. Dawid Górniak, Piotr Kopniak. Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska.

KRYPTOGRAFICZNE FUNKCJE SKRÓTU. Przemysław Rodwald. ZESZYTY NAUKOWE AKADEMII MARYNARKI WOJENNEJ ROK LIV NR 2 (193) 2013.