Student IT/EE Workshop 2025

Contribution ID: **57** Type: **Poster**

Comparison of the Performance and Effectiveness of Hashing Algorithms

Thursday 24 April 2025 11:30 (30 minutes)

\documentclass[conference]{IEEEtran}
\usepackage[T1]{fontenc}
\usepackage{lscape}
\IEEEoverridecommandlockouts
% The preceding line is only needed to identify funding in the first footnote. If that is unneeded, please comment it out.
\usepackage{cite}
\usepackage{amsmath,amssymb,amsfonts}
\usepackage{algorithmic}
\usepackage{graphicx}
\usepackage{textcomp}
\usepackage{xcolor}
\def\BibTeX{{\rm B\kern-.05em{\sc i\kern-.025em b}\kern-.08em
T\kern-.1667em\lower.7ex\hbox{E}\kern-.125emX}}
\begin{document}

\title{Comparison of the Performance and Effectiveness of Hashing Algorithms\\
}

\author{\IEEEauthorblockN{1st Konrad Žilinski}
\IEEEauthorblockA{\textit{Institute of Theory of Electrical Engineering, Measurement and Information Systems} \\
\textit{Faculty of Electrical Engineering}\\
Pl. Politechniki 1, 00-661 Warsaw, Poland \\
konrad.zilinski.stud@pw.edu.pl}
}

\maketitle

\begin{abstract}
This study compares the performance and efficiency of various hashing algorithms with a focus on their collision resistance and computational speed. The study was conducted on a mobile platform, analyzing historical and modern implementations of hash functions such as MD5, SHA-1, SHA-2, SHA-3, and Blake in different variants. The paper discusses their applications, advantages, and disadvantages, as well as the impact of architecture on performance. The results of the tests made it possible to identify the algorithm that offers the optimal compromise between security and computational efficiency.
\end{abstract}

\begin{IEEEkeywords}
hashing functions, hash collision, mobile platform, performance
\end{IEEEkeywords}

\section{Introduction}
Hash functions are used every day in most of the world's electronic devices. They are an indispensable part of cryptography, structures, and checksums. The following discusses the concept of hash functions and their use.

\subsection{Hash function, what it is? }
A hash function is a set of actions that unidirectionally transforms input data into output data of a certain length. An important aspect of such a function is, as mentioned earlier, the unidirectionality of the transformations. This means that the output data does not offer any information about the content of the input data. No less important for a mixing function is its sensitivity to changes in the input data. Although an ideal hash function should generate a different result with a change as small as one bit, it should also generate a completely unique result for each unique file.

\subsection{Where are they used?}
The simplest application of the hash function is indexing. It involves assigning a unique ID to any type of data.

When we need to arrange voluminous data in a structured way, it is inefficient and inconvenient to operate on entire files. Instead, the standard is to use mixing functions to assign unique strings of bits representing individual files. Thus, operations such as searches are greatly sped up because they use short strings.
Another realm in which hash functions are used is checksums. A hash function always produces the same result for the same input data. This property is used to confirm the immutability of the file. The result of the hash function is sent along with the file, and when it is received, another checksum is counted. If both checksums match, it means that the file has not been altered in any way.
However, the last but not least important application is cryptography. Hash functions are used for many different purposes. One of them is to make memorized passwords obfuscated. Instead of keeping passwords in the database, they are converted into the results of mixing functions. This ensures that passwords will not be disclosed during a potential data leak.

\section{Scope of the study}

No hash function is perfect. Each set of instructions compromises the speed of operation and quality of the result. In this study, I focused on the cryptographic part of hash functions. Their most important feature is resistance to result collisions, but this is not the only metric that will help us measure the performance of different functions.
A mobile platform was chosen to perform the study. The choice was based on several factors. First, mobile devices are widely available. The constant pursuit of the latest models causes us to accumulate many phones that are still fully operational. From this, the next reason for choice also arises - the historical aspect. This will allow us to compare the changes in trends in optimizing individual components.

\section{Tested functions}

\subsection {Message Digest (MD5)}
MD5 is the most popular mixing algorithm of the MD family. It was developed by Ron Rivers in 1991 as the successor to MD4. This algorithm generates hashes of 128 bits using the Merkle-Damgard architecture. However, the complexity of finding the collision of the result is not 2^{128} but 2^{18} \cite{b4}.

\subsection {Secure Hash Algorithm (SHA)}

\subsubsection {SHA-1}
In 1995, the National Institute of Standards and Technology (NIST) published a new hashing algorithm - SHA-1. Also, like MD5, it is based on the Merkle-Damgard architecture, and it only produces 160-bit hashes. And like its predecessors, it sweeps the ideal 2^{160} needed SHA-1 relies on at 2^{61} attempts.\cite{b4}

\subsubsection {SHA-2}
The next iteration of the Secure Hashing Algorithm hit the world in 2002. This time, the number of output bits is configurable, with 256, 384, and 512 bit variations available. SHA-2 is currently widely used, for example, in the Bitcoin cryptocurrency. However, like its predecessors using the Merkle-Damgard structure, hash collisions are increasingly common.

\subsubsection{SHA-3}
The next iteration of the Secure Mixing Algorithm hit the world in 2002. This time, the number of output bits is configurable, with 256, 384, and 512 bit variations available. SHA-2 is currently widely used, for example, in the Bitcoin cryptocurrency. However, like its predecessors using the Merkle-Damgard structure, hash collisions are increasingly common.

\subsection {BLAKE}
BLAKE is one of the algorithms that competed for SHA-3. Based on the ChaCha encryption string, BLAKE has several iterations and variants. BLAKE2s is designed to run on 32-bit systems, while BLAKE2b runs on 64-bit systems. BLAKE3 is the latest iteration, improving the algorithm's performance.

\section{Devices used in the study}

\begin{table}[htbp]
\caption{Information about the devices used in the study}
\begin{center}
\begin{tabular}{|c|p{0.16\linewidth}|c|p{0.11\linewidth}|c|}
\hline
\textbf{Manufacturer} & \textbf{Model Name}& \textbf{Model}& \textbf{Android version} & \textbf{Premier}\\
\hline\hline
Samsung & Galaxy A5 & SM-A500FU & 6.0.1 & 2014 Q4 \\
\hline
Samsung & Galaxy Tab A 7.0 & SM-T280 & 5.1.1 & 2016 Q1 \\
\hline
Sony & Xperia XZ Premium & G8141 & 9 & 2017 Q1 \\
\hline
Xiaomi & Mi A1 & Mi A1 & 9 & 2017 Q3 \\

```
\hline
Samsung & Galaxy S10e & SM-G970F & 12 & 2019 Q1 \\
\hline
Samsung & Galaxy A21s & SM-A217F & 12 & 2020 Q1 \\
\hline
Sony & Xperia 1 IV & XQ-CT54 & 14 & 2022 Q2 \\
\hline
Samsung & Galaxy A54 & SM-A546B & 14 & 2023 Q1 \\
\hline
\end{tabular}
\label{tab1}
\end{center}
\end{table}
```

```
\begin{table}[htbp]
\caption{Technical information about the devices used in the study}
\begin{center}
\begin{tabular}{|p{0.16\linewidth}|p{0.13\linewidth}|p{0.14\linewidth}|p{0.36\linewidth}|}
\hline
\textbf{Model Name}& \textbf{RAM} & \textbf{Chipset}& \textbf{CPU} \\
\hline\hline
Galaxy A5 & 2GB LPDDR3 & Snapdragon 410 & 4x Cortex-A53 1190MHz \\
\hline
Galaxy Tab A 7.0 & 1.5GB LPDDR3 & Spreadtrum SC7730S & 4x Cortex-A7 1300MHz \\
\hline
Xperia XZ Premium & 4GB LPDDR4X & Snapdragon 835 & 4x Qualcomm Kryo 280LP 1900MHz, \newline 4x Qualcomm Kryo 280HP 2457MHz \\
\hline
Mi A1 & 4GB LPDDR3 & Snapdragon 625 & 8x Cortex-A53 2016MHz \\
\hline
Galaxy S10e & 6GB LPDDR4X & Samsung Exynos 9820 & 2x Mongoose M4 2730MHz,\newline 2x Cortex-A75 2310MHz,\newline 4x Cortex-A55 1950MHz \\
\hline
Galaxy A21s & 3GB LPDDR4X & Samsung Exynos 850 & 8x Cortex-A55 2002MHz \\
\hline
Xperia 1 IV & 12GB LPDDR5 & Snapdragon 8 Gen 1 & 4x Cortex-A510 1785MHz, \newline 3x Cortex-A710 2495MHz, \newline Cortex-X2 2995MHz \\
\hline
Galaxy A54 & 8GB LPDDR4X & Samsung Exynos 1380 & 4x Cortex-A55 2002MHz, \newline4x Cortex-A78 2400MHz \\
\hline
\end{tabular}
\label{tab1}
\end{center}
\end{table}
```

Tables 1 and 2 show all the information about devices used in the study. These devices are from 3 companies: SONY, SAMSUNG, and XIAOMI. The oldest tested phone premiered more than 11 years ago. While the newest - barely 2. All these phones are from different price shelves, from budget variants to flagship models.

\section{The application used to conduct the study}

Virtually every device uses a different version of Android, so a key aspect of the study was to choose a solution that would allow a comparison of all devices across 10 years of development. The first idea was to use the Termux application, allowing a .jar file to run. However, it was not possible to provide the same version of Java for all devices, more specifically, the oldest ones. The logical next step was to abandon running the files directly and create an entire mobile application. This is a simple application created using Android Studio. As you can see in "Fig.~\ref{rys:applikacja}", it takes 3 buttons. 1 - start the calculation, 2 - start the calculation in reverse order, and 3 - export the results to a .csv file. The calculation phase is started with a warm-up round, which greatly improves the consistency of results between runs. Then, 100 executions of mixing functions are divided into the available number of device cores. A data buffer is created for each new mixing function execution and for each data size. Time is measured only from the start of hashing to the receipt of output data.

```
\begin{figure}[htbp]
\centerline{\includegraphics[width=0.9\linewidth]{images/screenshot.png}}
\caption{Application screenshot}
```

\label{rys:applikacja}
\end{figure}

\section{Course of the study}

The first step of the study was to prepare the devices. Each device was charged and rebooted, excess files in memory were removed, and also airplane mode was turned on. Before each app launch, I made sure the device had a battery level above 40 and also that it was not hot. As part of the test, the app was run 3 times, twice forward and once backward, to average out any variation caused by device heating. Also, during the test, the device's display was periodically activated so that the phone did not fall into sleep mode. After each activation, the data was saved in the device's memory and then copied to a computer for analysis.

\section{Results}

\subsection{Simplest comparison}
The most obvious comparison that can be made is to create a time diagram of all calculations for a particular device. An example of a diagram from a Xiaomi Mi A1 device can be seen in "Fig.~\ref{rys:wykresGlownyXiaomi}". It clearly shows that the fastest functions are SHA-1 and SHA-256. Next comes MD5, and right behind it are SHA-512 and SHA-384. Next, algorithms from the SHA3 series are ordered. The tables are closed with algorithms from the BLAKE2 and BLAKE3 series.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/xiaomi.png}}
\caption{Average calculation time for Xiaomi Mi A1}
\label{rys:wykresGlownyXiaomi}
\end{figure}

\subsection{Historical context}

The obvious next comparison is to add historical context. "Fig.~\ref{rys:zmiennoscWCzasie}" compares the computation time of 3 hashing functions (MD5, SHA1, and BLAKE2b) for two input data lengths(32MB and 32KB) on all devices. First, of course, all calculations on 32KB are faster than 32MB. Next, surprisingly, the MD5 is the fastest only on Samsung's oldest device. It is also interesting to note that a device 2 years younger (Samsung Tab A) is significantly faster at calculating SHA-3 and BLAKE2b but slower at calculating SHA1 and MD5. One more thing to mention is the distinct separation between premium models and budget ones. All prices models have multi architectural CPUs that allow the to excel in specific areas, On older Samsung phone from 2019 it can be observed particularly well as SHA3 caclucaltions were significantly decreased in time more than SHA1.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/zmiennoscWCzasie.png}}
\caption{Time comparison for particular hash functions for different devices}
\label{rys:zmiennoscWCzasie}
\end{figure}

\subsection{Moores Law}

Fig.~\ref{rys:zmiennoscWCzasie}" shows that Moore's law is not preserved. The time required to calculate the hash function does not decrease twice yearly.Fig.~\ref{rys:newvsold}" shows the acceleration for all types of functions. The average acceleration over 10 years is 14.6 times. This result was obtained by comparing the oldest mid-priced phone with the newest mid-priced phone from the same manufacturer, Samsung. Changing the price segment to premium, on the other hand, outlines significantly different values. The 5 years of development in this segment accelerates the performance by 20 percent alone.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/newvsold.png}}
\caption{Improvement comparison for different hash functions}
\label{rys:newvsold}
\end{figure}

\subsection{Comparison against the fastest function}

The previous diagrams show the execution time of various functions changing over the years. However, it illustrates not only the improvement of the algorithm but also the speeding up of the devices themselves.
On all devices except the oldest, the fastest hash function was the SHA-1 function.
It was chosen as the baseline.

\subsubsection{SHA-1 VS SHA-3}

Fig.~\ref{rys:sha1vsSha3}" shows how much faster SHA-1 is compared to SHA-3. On the graph, we can identify 3 main groups. The slowest SHA-3 is on the oldest device, which is expected, but surprisingly, the second worst one is a phone from the year 2020. Next is a group of budget phones, and the last group is for the newest or premium phones. The most important part of this diagram is the results for the device premiered in 2017 Q3. This indicates that although this phone is one of the fastest (Fig.~\ref{rys:zmiennoscWCzasie}"), its performance is not related to the acceleration of the SHA-3 algorithm but plain hardware improvement.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/sha1vssha3.png}}
\caption{SHA-1 vs SHA-3}
\label{rys:sha1vsSha3}
\end{figure}

\subsubsection{SHA-1 VS BLAKE}

Different BLAKE hashing algorithms have similar results. There are small differences that we were able to measure, from which it can be concluded that BLAKE in version 3 is the slowest one. Generally, BLAKE2b-256 is the best performer, but it is always slower than SHA-3, even compared to the 512-bit variant. Worth noting here is that 512-bit variants are marginally slower than their 256 versions, which indicates that these functions are not optimized for smaller sizes. BLAKE2s, which is meant for 32-bit devices, is also not optimized enough to make any significant impact - on devices that use 32-bit architecture, this algorithm matches the performance of BLAKE2b, and on all other devices, it performs worse. Comparison of BLAKE algorithms to SHA-1 is not much different from the SHA-3 situation: the worst performers are the oldest phone and the budget one.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/sha1vsBlake2b256.png}}
\caption{SHA-1 vs BLAKE2b-256}
\label{rys:sha1vssha3}
\end{figure}

\subsubsection{SHA-1 VS SHA-256}

SHA-256 also known as SHA2 is still widely used algorithm. Surprisingly, in the majority of devices, it performed very similarly to the deprecated SHA-1 algorithm, providing higher security. Based on "Fig.~\ref{rys:sha1vssha2}" it can be easily noted that SHA-1 performs up to 100\% better than SHA-256 on older phones. Subsequently, the newer the device, the better SHA-256 gets, matching or outperforming slightly SHA-1. Also, it is important to mention that differences here are measured in percentages because results are not bigger than twice the size. In contrast, differences with the BLAKE algorithm reached 60000\%.

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/sha1vssha2.png}}
\caption{SHA-1 vs SHA-256}
\label{rys:sha1vssha2}
\end{figure}

\subsection{Collision resolution context}

All previous comparisons were based on measured time. Some hash functions were faster, some slower. As stated before, calculation speed is only one of the factors by which we can measure hash quality. An equally important aspect is also how collision-resistant the tested function is. To properly illustrate performance, taking into account both hashing time and offered collision resistance, equation \eqref{eq}" was developed. To aggregate further results, SHA-1 was once again used as a reference point, resulting in graphs-Fig.~\ref{rys:overallnew}" and "Fig.~\ref{rys:overallold}".

\begin{equation}
\frac{TrueCollisionResistance / MaxCollisionResistance}{TimeTaken} \label{eq}
\end{equation}

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/finalold.png}}
\caption{Relative overall hash function performance for old device}
\label{rys:overallold}
\end{figure}

\begin{figure}[!hb]
\centerline{\includegraphics[width=0.9\linewidth]{images/finalnew.png}}
\caption{Relative overall hash function performance for newest device}
\label{rys:overallnew}
\end{figure}

Both of these graphs clearly indicate that the best performance is achieved by SHA-2 algorithms. It provides the biggest collision resistance quickest across all device generations. Comparing old and new devices, we also can conclude that the SHA-3 algorithm is catching up with the leader. It is close to outperforming SHA-384 and SHA-512. Second in line are BLAKE family algorithms. They provide similar collision resistance as SHA-3 but take significantly longer to calculate, resulting in lower performance. SHA-1 and MD5 did not make it onto the podium because of their lack of vulnerabilities. They are the fastest, but they also are not cryptographically viable.

\section{Overview}

The study comprehensively compared the performance and effectiveness of popular hashing algorithms across a diverse set of mobile devices spanning nearly a decade. It revealed that while older algorithms like MD5 and SHA-1 still demonstrate high speed, their lack of collision resistance renders them unsuitable for secure applications. Modern algorithms, especially SHA-2, strike the best balance between performance and cryptographic strength, maintaining high resistance to collisions with relatively fast processing times. Although SHA-3 and BLAKE series offer comparable security, their performance is notably dependent on device hardware and architecture, which limits their efficiency on budget or older devices. The findings confirm that algorithmic performance scales with both time and hardware capabilities, but not always proportionally, indicating that optimization plays a key role. Analysis proves that we should use tested methods like SHA-2 and not blindly use new and shiny ones, as paper studies \cite{b3} are sometimes far away from actual results.

\begin{thebibliography}{00}
\bibitem{b1} ALAMGIR, Nahiyan; NEJATI, Saeed; BRIGHT, Curtis. SHA-256 collision attack with programmatic SAT. arXiv preprint arXiv:2406.20072, 2024.
\bibitem{b2} GAURAVARAM, Praveen; MCCULLAGH, Adrian; DAWSON, Edward. Collision attacks on MD5 and SHA-1: Is this the "Sword of Damocles"for electronic commerce. Information Security Institue (ISI), Queensland University of Technology (QUT), Australia, 2006.
\bibitem{b3} BLACK, John; COCHRAN, Martin; HIGHLAND, Trevor. A study of the MD5 attacks: Insights and improvements. In: Fast Software Encryption: 13th International Workshop, FSE 2006, Graz, Austria, March 15-17, 2006, Revised Selected Papers 13. Springer Berlin Heidelberg, 2006. p. 262-277.
\bibitem{b4} MAETOUQ, Ali, et al. Comparison of hash function algorithms against attacks: A review. International Journal of Advanced Computer Science and Applications, 2018, 9.8.
\bibitem{b5} AUMASSON, Jean-Philippe, et al. BLAKE2: simpler, smaller, fast as MD5. In: International Conference on Applied Cryptography and Network Security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 119-135.
\bibitem{b6} SASAKI, Yu, et al. Improved collision attack on MD5. Cryptology ePrint Archive, 2005.
\bibitem{b7} DEN BOER, Bert; BOSSELAERS, Antoon. Collisions for the compression function of MD5. In: Workshop on the Theory and Application of of Cryptographic Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993. p. 293-304.

\end{thebibliography}
\vspace{12pt}

\end{document}

Author: ZILINSKI, Konrad

Presenter: ZILINSKI, Konrad

Session Classification: Session C (Poster)