Characterizing the Impact of Dynamic Resource Management on Message Authentication in Mobiles

A. M. Rashwan[1], A-E M. Taha[2], and H. S. Hassanein[1]

[1]Telecommunications Research Lab
School of Computing
Queen's University
Kingston, ON, Canada K7L 3N6
{arashwan, hossam}@cs.queensu.ca

[2]Electrical Engineering Department
Alfaisal University
P.O. Box 5092
Riyadh 11533 KSA
ataha@alfaisal.edu

***Abstract*— Communication security measures are becoming prominent standards of next generation mobile networks. These measures usually involve resource-intensive cryptographic operations that can greatly affect communication performance, particularly when it comes to time-based guarantees such as delay and jitter. It this becomes vital to understand the computational characteristics of such operations from a communication perspective. The determination of these characteristics, however, is challenging with mobile computing as computational resources are dynamically managed for balanced performance and energy-efficiency. This paper investigates different mobile computing resource management features, and evaluates their impact on the evaluation of cryptographic functions used in message authentication, and on the design considerations for future context-aware security protocols.**

Keywords-component; dynamic resource management benchmarking; mobile computing security; message authentication code; message hashing; next generation Internet security.

I. INTRODUCTION

Today's mobile computing systems incorporate sophisticated features intended to improve performance while saving energy through dynamically managing the processors clocking speed and utilization. In addition, mobile processors have some sort of thermal protection, intended primarily to protect the processors from overheating by shutting/slowing them down. Moreover, mobile computing systems usually operate on batteries, which their lifetime can be affected greatly by the heat generated from the systems' processors. Therefore, vendors developed enhanced thermal management techniques that controls both speed and utilization features in quest of achieving better balance between the performance and the battery lifetime of the mobile systems.

Applying security measures is an integral element of communications in the next generation Internet. As such, and given their computationally-intensive operation, it becomes essential to study how security operations will behave on a dynamically-managed environment such as in mobile systems. Such study can help optimizing the security operations for dynamic mobile environments while achieving both acceptable security and performance measures, especially in terms of time-based Quality of Service (QoS) metrics such as delay and jitter.

This paper investigates the effect of controlling the aforementioned mobile computing features on the behavior of some selected cryptographic functions. In this paper, we focus on the cryptographic functions that generate *Message Authentication Code* (MAC); a security measure aimed to provide data integrity protection service for communication sessions. In this paper, the computational characteristics of the selected functions are observed through benchmarking them from a communication perspective; under selected mobile computing architectures. To facilitate a meaningful comparison in the communication context, we use the *apparent processing power*; a metric for evaluating the computational complexity as seen by initiating mobile systems.

The remainder of this paper is organized as follows. Section II refers to the background and motivations behind inducting the study of the mobile dynamic resource management on the computational characteristics of MAC functions. Section III describes the benchmarking procedure applied in the study. Evaluation of the benchmarking results is presented in Section IV. Finally, conclusion and future directions are mentioned in Section VI.

II. BACKGROUND AND MOTIVATION

Securing network communications include applying measures at different levels to ensure confidentiality, authenticity, and integrity of the transmitted information. One of the known measures used in communication security involves applying of message authentication codes, or MACs, which are secured verification codes attached to their corresponding messages to ensure their data integrity. MACs are usually generated by some sort of one-way cryptographic functions, where protected messages cannot be easily reconstructed giving their corresponding MAC.

As an effective measure to ensure information integrity, MAC functions had found their ways into various security protocols implemented today such as IPSec, SSL/TLS, Kerberos, TCP-AO, and SCTP-AUTH, to name a few [1]. These protocols were usually implemented with predetermined MAC functions. Moreover, some of these protocols were designed to allow the selection of different MAC functions in quest of meeting certain security requirements. However, these protocols were not designed to handle the impact of having communication systems with diverse resource capabilities on their operating behaviors. Without such handling, the performance of these protocols, represented normally by the latency, jitter, and throughput from a communication perspective, is not manageable. Such lacking can lead to undesirable communication and system performance issues, especially in next generation networks such as *Internet-of-Thing* (IoT) and *Information-Centric-Network* (ICN), where many of the communicating systems are mobile and with controlled/limited resource capabilities.

978-1-4799-2003-7/14/$31.00 ©2014 IEEE

From a communication perspective, the resource capability does not only point to the absolute computational power required from the mobile system, which represents a main focus of evaluating cryptographic functions in the literature [2] [3][4][5][6][7]. Determining effective resource capabilities can also be subject to the following factors:

- **Context.** The resource capability can be a reflection to the context of the surrounding environment. For example, factors such as the operating temperature, power source, local time, activities of other communicating entities on the local network, etc.; can greatly affect the processing timeline of the applied security protocols, regardless of the actual resources required. Thus, in communications, it is more informative to be aware of how much time is needed to apply communication security measures instead of how much resources will take. Such awareness can be helpful for more effective Quality-of-Service (QoS) management in communications guarantees, which is mainly correlated to the transmission latency, not to the absolute resource utilization.

- **Implementation Heterogeneity.** The resource capability can also refer to the implementation of the MAC functions (and the security protocols in general) on the communicating systems. Some systems may have hardware circuitry or hardware-assisted instructions for the cryptographic or hashing MAC engines. Some systems use MAC functions that are designed to run on multiple simultaneous processors, while other systems use functions that are designed to run on single processors. Even systems of same physical capabilities can have different software implementations of same MAC functions. Such implementation-dependency leads into diverse performance trends between communicating systems. This diversity can cause various performance or availability issues for limited-resource communicating systems, such as sensors, and systems with resource-demanding MAC function implementations. For example, a system with resource-demanding implementation of a certain function can be overwhelmed, and eventually become unavailable, while communicating with another system that has a resource-optimized implementation of the same function.

- **Resource Management.** Most existing mobile systems are equipped with some sort of a resource management technique, aimed mainly at providing an energy-efficient operation. In such environment, the useful resource capabilities are subject to the mobile system's resource management, which operates dynamically based on the processing demands. As with the implementation heterogeneity, communications between mobile systems with dynamic resource managements can have diverse performance trends; including the non-deterministic performance expectations of MAC functions operating on those communications. In addition, expectations of how mobile system resources are controlled does not necessarily reflect on how MAC functions will behave. For example, doubling the processor clocking speed on a mobile system does not mean that the operating MAC functions will take exactly half the processing time, as they are subject to other architecture factors such as the design of the processor execution pipeline, and the I/O bus' speed.

Based on the aforementioned factors, just relying on analytical or absolute evaluations of cryptographic functions is not beneficial or practical from a communication perspective. It is clear that sticking with predetermined MAC functionality (as with the existing security protocols) does not suite every mobile system; mainly due to their diversity in term of resource capabilities. Moreover, relying on having a predetermined MAC or other cryptographic functionality during a connection lifetime makes it vulnerable for easier attacks as opposed for having frequent changing of the applied cryptographic engine during that connection lifetime. Therefore, future generation of security protocols may need to implement some sort of adaptive selection and operating mechanisms that securely serve current and future communicating systems regardless of their capabilities.

An adaptive security protocol may not perform effectively, from a communication perspective, with only the knowledge of analytical or absolute evaluations of its underlying cryptographic functions. It is also important to be aware of the factors that control the resource capabilities of communicating systems, and to evaluate their effect on the performance trend of the cryptographic functions. Part of this evaluation was introduced in [8]. The work presented herein, however, is aimed at studying how the dynamic resource management of mobile systems can affect their resource capabilities and so the evaluation of MAC functions. The focus in this paper is to investigate the effect of certain mobile computing features on the keying and tagging performance of evaluated MAC functions.

III. THE BENCHMARKING PROCEDURE

Conducting an evaluation in a dynamic computing environment is very challenging and does not usually provide informative results. For an effective investigation of the effect of such environment on the evaluation of MAC functions, it is important to isolate the mobile computing features that are directly related to the execution time or characteristics. Such isolation allows for having a benchmarking environment that has better control on obtaining useful evaluation from mobile systems with dynamic resource management, yet keeps the nature of the mobile environments as realistic as possible.

In this paper, the benchmarking setup in [8] is substantially altered, as shown in Fig. 1, to fulfill the aforementioned objective. First, the following mobile computing features, which were found to be related to the execution time or characteristics, are identified:

- *Frequency Scaling*: aims to reduce or increase processor's clocking speed based on the applications' demands.
- *Simultaneous hardware Multi-Threading* (SMT): aims to improve processor utilization (and reduce wasted energy from underutilization) through executing multiple hardware threads per core. With SMT enabled, it is expected that the processor temperature to be higher as a result of working harder in such case.
- *Processor/Core Parking*: aims to park (aka shutdown or idle) some of the active processor/cores in case of low demands. Modern mobile systems can also utilize this feature in order to reduce the operating temperature.

A module to control the aforementioned features is integrated to the benchmarking setup and the workload is accordingly adjusted. The newly integrated module allows the

application of dynamic operating environments and to obtain meaningful results in a controlled manner. The evaluation metrics from [8] are selected with slight variations to fit the requirements of this study.

The following details the elements of the benchmarking setup.

A. Environment

All benchmarking experiments were conducted under the Linux operating system environment (Ubuntu 12.04 LTS in this study). Moreover, the well-known cryptographic library CryptoPP (v. 5.6.2) [5] is used for this study as the provider for the evaluated MAC functions. All evaluated MAC functions are complied with no hardware-assisted optimizations.

Evaluation results were obtained for MAC functions running under the following architectures that represent most of the existing mobile systems in the market:

- *x86-based (32-bit) laptops and tablets:* Intel Core I5 650 (32-bit mode; Dual-core with SMT), Intel Pentium 4 M 3.0GHz (Single-core with SMT), Intel Atom D525 (Dual-core with SMT), and AMD Opteron 2354 (32-bit mode; Quad-core)
- *x86-based (64-bit) laptops and tablets:* Intel Core I5 650 (64-bit mode; Dual-core with SMT)
- *ARM-based (32-bit) smart phones and tablets:* Texas Instruments' DM3730 ARM Cortex™ A8 (Single-core), and Texas Instruments' OMAP 4460 ARM Dual-core Cortex™ A9 (Dual-core)

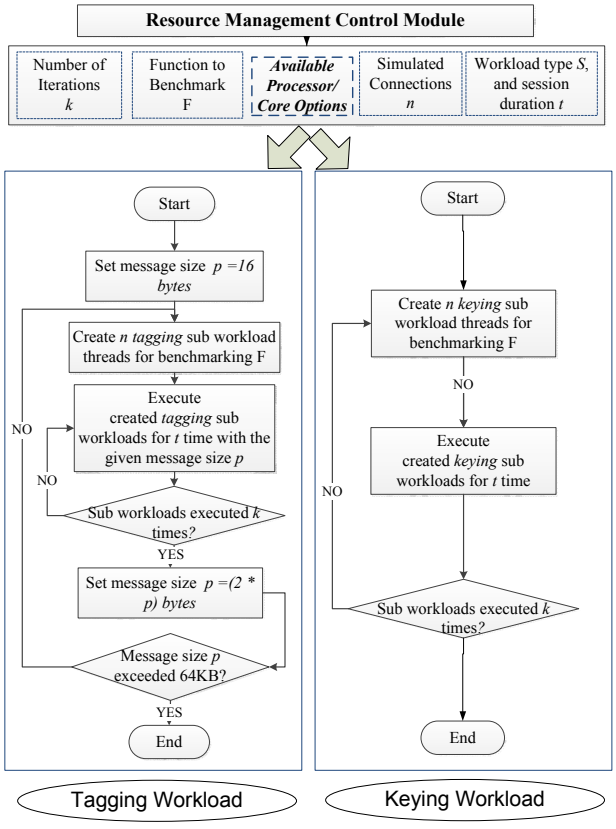


Fig. 1. Benchmark main workload procedure

B. Metrics

MAC functionality involves three key operations. The first operation is known as *keying*, where a secret key is assigned to the MAC function to be used in generating secure MACs. The second operation is known as *tagging*, where MACs are generated and attached to the corresponding messages. The third operation is known as *validating*, where a message is validated against a MAC attached to it.

In this study, only "keying" and "tagging" operations are considered for the evaluation, as the "validating" operates similarly to the "tagging" operation.

- For the tagging operation:
  - *Apparent "tagging" Processing Power*: The estimated cycles (as they appear to the calling mobile system) that a CPU uses to execute tagging instructions to process one byte of the given message (computed from the mobile system wall-based clock).
  - *Absolute "tagging" Processing Power*: The actual cycles that the CPU uses to execute tagging instructions to process one byte of data (obtained from mobile system processor clock).
- For the keying operation:
  - *Apparent "keying" Processing Power*: The estimated cycles (as they appear to the calling mobile system) that a CPU uses to execute keying instructions to setup a new secret key (computed from the mobile system wall-based clock).
  - *Absolute "keying" Processing Power*: The actual cycles that the CPU uses to execute keying instructions to setup a new secret key (obtained from mobile system processor clock).

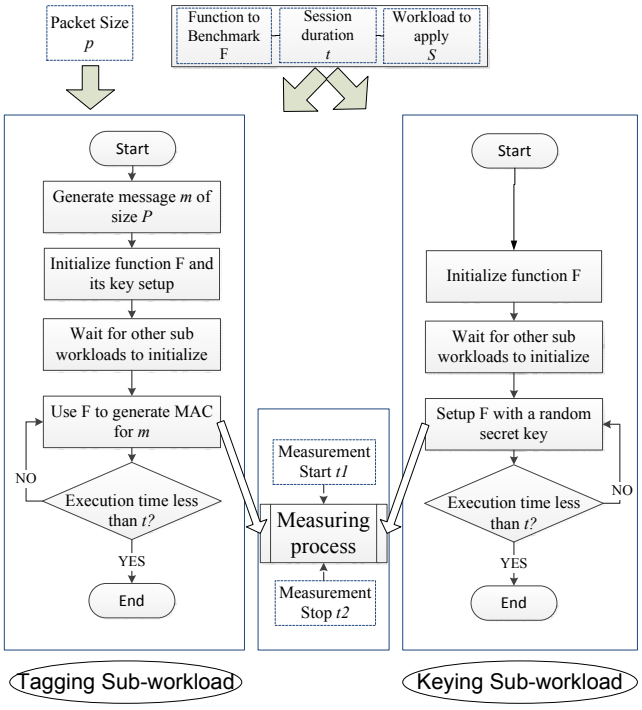


Fig. 2. Benchmark sub workload procedure

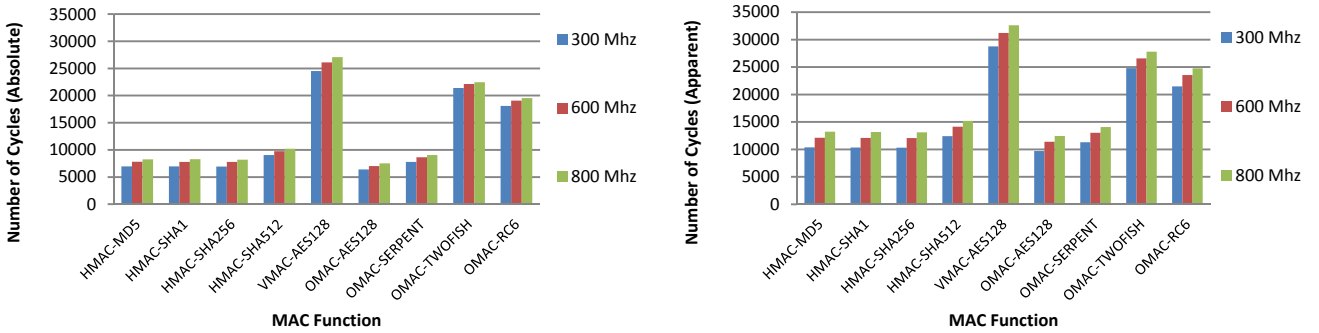


Fig. 3. Processing Power of Keying Operations verses CPU frequency under Texas Instruments' DM3730 ARM Cortex™ A8 (32-bit mode)

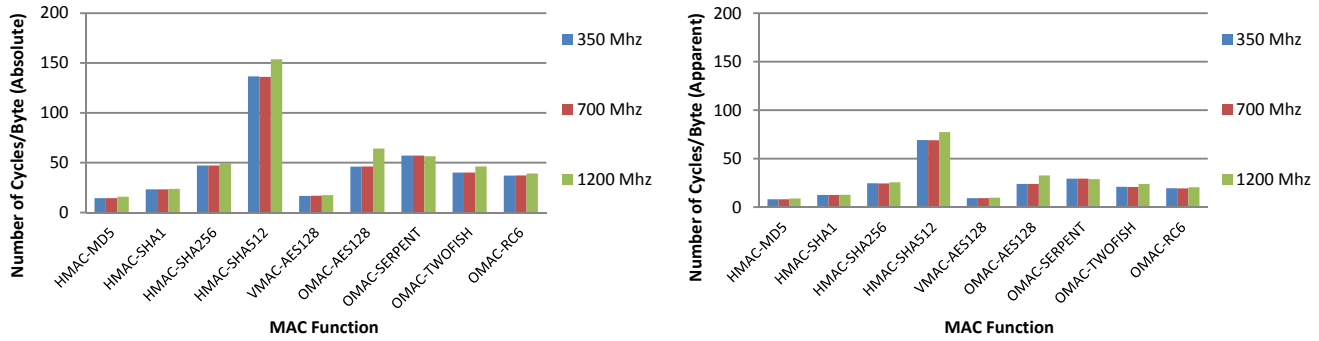


Fig. 4. Processing Power of Tagging Operations (2048-byte messages) verses CPU frequency under Texas Instruments' OMAP 4460 ARM Dual-core Cortex™ A9 (32-bit mode)

C. *Workload*

The multi-threaded benchmarking application includes the aforementioned resource control module that provides a controlled-version of the dynamic resource management to be evaluated. With the help of this module, the main workload procedure is provided with different resource management scenarios. These scenarios consist of applying different operating clock frequencies, enabling and disabling the SMT functionality, and idling one of more of the available processors/cores. The main workload procedure then applies the experiments based on the given scenario.

The main workload and the sub workload allow for the evaluation of both the tagging and the keying operations as shown in Figs. 1 and 2. Similar to the tagging sub workloads, the measurements in keying sub workloads are also captured using both the processor clock and the system's wall-based clock. Unlike the tagging operation, the keying operation does not depend on the message size.

D. *Measuring Assumptions*

The following are not taken into consideration when conducting this study:

- Processing memory utilization (assumed to be reflected on the processing power to some degree due to the increased I/O activity resulted from the process/thread transfer into memory for sleeping).
- Voltage scaling: where power is saved by lowering the operating (core) voltage. This mobile feature is not manually controllable for the sake of processor stability. However, we expect that voltage scaling should not affect processing power by any mean.

IV. EVALUATION

A. *The Effect of Controlling Frequency Scaling*

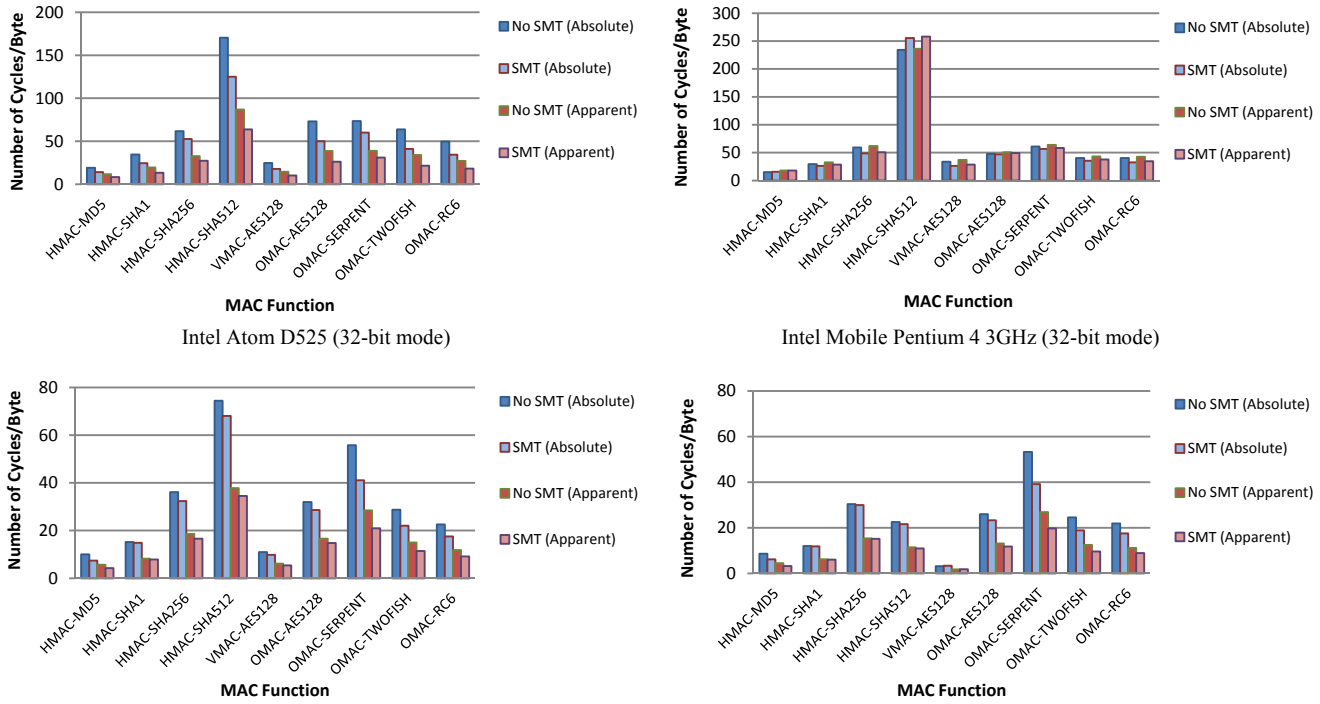
The results showing the effect of controlling frequency scaling are shown in Fig. 3 and Fig. 4.

It is intuitive to expect that the absolute processing power should remain the same regardless of the changes made in frequency scaling (However, the actual processing time is naturally expected to change). This is because absolute processing power is measured in cycles, not seconds. Practically, however, this is not the case for all/most evaluated processors.

The only benchmarked system that follows the aforementioned intuition is the Intel Core I5 650, which exhibits only slight irregularities in the performance of keying operations. These irregularities are mainly due to the OS scheduling and background processes contention on resources, especially with the keying operations being of the light-processing type.

In architectures with considerably slow memory and I/O access compared to their processor clocking speed (especially with entry-level systems such as ARM architectures with high-speed CPU clocks), increasing the processor clocking frequency leads into an increased number of wasted cycles. This appears noticeably with light-processing (Fig. 3) and/or I/O-intensive (Fig. 4) operations, such as with HMAC-SHA512 in 32-bit mode which requires extensive memory access.

Generally, applying the frequency scaling has resulted into non-deterministic computational behaviors for the benchmarked tagging/keying operations. Some tagging operations, such as in OMAC-SERPENT, have mostly showed



Intel Atom D525 (32-bit mode)

Intel Mobile Pentium 4 3GHz (32-bit mode)

Intel Core I5 650 (Left: 32-bit mode, Right: 64-bit mode)

Fig. 5. The effect of enabling SMT on the tagging processing power (2048-byte messages) of selected architectures

slight performance improvements with the increased clocking speed regardless of benchmarked architectures and with no determination found between the operating frequency and processing power. HMAC-MD5 tagging operation, on the other hand, has shown slight performance degradation with the increase clocking speed across the benchmarked architectures.

B. *The Effect of Enabling/Disabling SMT*

We observe that enabling SMT generally improves the performance of tagging/keying operations; up to around 30% as shown in Fig. 5, since the resources of the SMT-enabled processors, by design, are utilized more effectively. However, some evaluated systems, such as Intel Pentium 4, show performance degradation among all evaluated MAC functions. This can be due to the lack of sufficient instruction caching space; causing excessive cache misses that reflects on the SMT poor performance [9]. Generally, the performance improvement/degradation is system-dependent as shown in Fig. 5. The degree of improvement/degradation is subject mainly to how the system processor is designed to handle instruction caching.

C. *The Effect of Core Parking*

Fig. 6 shows the effect of core parking on the apparent processing power of selected architectures, with the percentage axis reflecting the additional cycles that the evaluated functions require after enforcing the parking. Although most of the evaluated functions show performance degradation, some show performance gain especially when the parked component is SMT related (as with VMAC-AES128 tagging operation in Intel Core I5 64-bit mode); which is due in turn to the effect of the excessive cache missing on the SMT performance.

It is also noted that the performance degradation trend caused by core parking is subject to the architecture of the evaluated system. For example, OMAC-AES128 has shown 1.5x performance degradation on an ARM-based architecture with one active core, while the same function has shown 2x performance degradation on x86 architectures with one active core.

D. *Apparent vs. Absolute Processing Power*

It is observed, from Figs. 5 and 6, that both apparent and absolute processing powers appear to somehow reflect the I/O demands and the memory space complexity. For example, HMAC-SHA512 requires higher I/O operations to handle its larger 1024-bit message blocks as opposed to 512-bit message blocks in the other evaluated functions. In the 64-bit mode, HMAC-SHA512 has benefited from the larger 64-bit register space which reflected positively on its I/O operations and so its performance.

We remark that only the apparent processing power has reflected the effect of core parking (with except to the SMT). This is expected since the absolute power represents the actual processing done by the processor. The apparent power is also appeared to more effectively reflect the operation's waiting/sleeping times accessing slow memory and I/O, especially if that operation does not require extensive processing power such as keying (Fig. 3).

V. CONCLUSIONS

In this paper, we evaluate the impact of dynamic resource management in mobile systems on the computational characteristics of cryptographic message authentication functions. We apply a tailored benchmarking procedure to

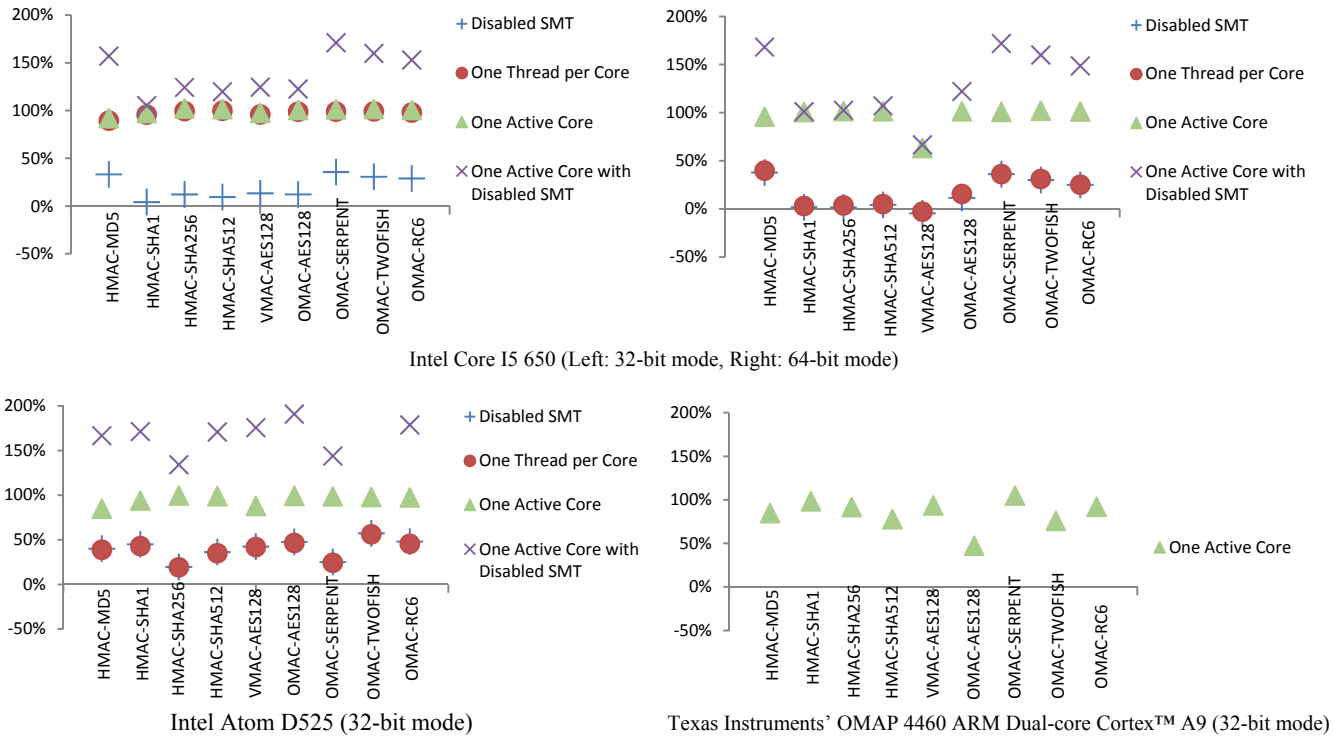


Fig. 6. The effect of parking on the performance degradation of the apparent tagging processing power (2048-byte messages) of selected architectures

observe the effects of resource management features, e.g. frequency scaling and core parking, on the performance trends of the evaluated functions. Based on the results, we observe various non-intuitive performance behaviors that adversely disrupt QoS performance, especially when it comes to time-based guarantees. For example, frequency scaling improves performance in certain architectures, while degrades performance in others. Meanwhile, SMT is not found to improve performance in certain architectures.

These and other findings indicate necessitate revising how MAC functions are selected in mobile platforms. Specifically, a need exists for applying novel MAC evaluation and selection mechanisms when it comes to communicating systems. The results presented herein can help establishing an adaptive functionality that ensures QoS based on the context of the communicating systems and the minimum service level requirements.

Finally, it is important to note that the absolute resource metrics, commonly used in works evaluating existing cryptographic functions, are not delay-based and do not reflect actual communication demands. Existing metrics, such as the absolute computational or throughput performance further do not reflect many of today's mobile computing sophisticated features and operational aspects. Having an evaluation metric such as the apparent performance, which is purposefully based on showing processing delays over actual processing performance, can be more informative from a communication perspective. Considering the simplicity of obtaining such metric, it can be a good candidate for future security protocols to provide an effective evaluation and adaptation of their functionalities.

ACKNOWLEDGMENT

This research is funded by a grant from Natural Sciences and Engineering Research Council of Canada (NSERC).

REFERENCES

[1] Internet Official Protocol Standards. RFC 5000, IETF, May 2008, http://www.ietf.org/rfc/rfc5000.txt.

[2] M. Sokol, S. Gajewski, M. Gajewska, and L. Staszkiewicz, "Security and Performance Analysis of IPsec-based VPNs in RSMAD," in *The First International Conference on Advanced Communications and Computation (INFOCOMP'11)*, 2011, pp. 70-74.

[3] J. Kaps and B. Sunar, "Energy Comparison of AES and SHA-1 for Ubiquitous Computing," in *Embedded and Ubiquitous Computing*, 2006, pp. 372-381.

[4] B. Schneier and D. Whiting, "A performance comparison of the five AES finalists," , 2001, p. the 3rd AES Conference (AES3).

[5] Y. W. Law, J. Doumen, and P. Hartel, "Survey and Benchmark of Block Ciphers for Wireless Sensor Networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 2, pp. 65-93, February 2006.

[6] D. S. Abd Elminaam, H. M. Abdual Kader, and M. M. Hadhoud, "Evaluating The Performance of Symmetric Encryption Algorithms," *International Journal of Network Security*, vol. 10, no. 3, pp. 216–222, 2010.

[7] O. Hyncica, P. Kucera, P. Honzik, and P. Fiedler, "Performance evaluation of symmetric cryptography in embedded systems," in *IEEE 6th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*, 2011, pp. 277-282.

[8] A. M. Rashwan, A. M. Taha, and H. S. Hassanein, "Benchmarking message authentication code functions for mobile computing," in *IEEE Global Communications Conference (GLOBECOM)*, Anaheim, California, 2012, pp. 2585-2590.

[9] Intel Corp. (2003, January) Intel® Hyper-Threading Technology Technical User's Guide. [Online]. http://cache-www.intel.com/cd/00/00/01/77/17705_htt_user_guide.pdf