12-2010

Time cost evaluation for executing RFID authentication protocols

Kevin CHIEW
Singapore Management University, kevinchiew@smu.edu.sg

Yingjiu LI
Singapore Management University, yjli@smu.edu.sg

Tieyan LI
Institute of Infocomm Research

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Manfred AIGNER
Technische Universitat Graz

DOI: https://doi.org/10.1109/IOT.2010.5678437

Time Cost Evaluation for Executing RFID Authentication Protocols

Kevin Chiew, Yingjiu Li
Singapore Management University, Singapore
{kevinchiew, yjli}@smu.edu.sg

Tieyan Li
Institute of Infocomm Research, Singapore
litieyan@i2r.a-star.edu.sg

Robert H. Deng
Singapore Management University, Singapore
robertdeng@smu.edu.sg

Manfred Aigner
Graz University of Technology, Austria
manfred.aigner@iaik.tugraz.at

***Abstract*—There are various reader/tag authentication protocols proposed for the security of RFID systems. Such a protocol normally contains several rounds of conversations between a tag and a reader and involves cryptographic operations at both reader and tag sides. Currently there is a lack of benchmarks that provide a fair comparison platform for (a) the time cost of cryptographic operations at the tag side and (b) the time cost of data exchange between a reader and a tag, making it impossible to evaluate the total time cost for executing a protocol. Based on our experiments implemented on IAIK UHF tag emulators (known as DemoTags), in this paper we present detailed benchmarks of both time costs for RFID authentication protocols. Our results reveal that the data exchange dominates the time cost for running a protocol. We also give a classification for the existing protocols and summarise formulae of time cost evaluation for each type of protocols in a way so that a user can evaluate the time cost of a protocol amongst several choices of cryptographic operations in an application.**

I. Introduction

The RFID (radio frequency identification) technology has been envisioned to have significant impact on the economy world-wide as an inevitable replacement of barcodes in the near future [1]. One of its most promising applications is supply chain management [2], [3], in which RFID tags are attached to products so that they can be conveniently identified by tag readers. An RFID applications may encounter insecure situations like duplication of tag IDs, invalid or counterfeit tags and readers, or other malicious attacks. Therefore, in practice it is critical to have solutions that can authenticate the legitimacy of tags and readers. Aiming at this purpose, various reader/tag authentication protocols [4]–[8] are proposed to enhance the system security, reliability, and robustness. Such a protocol normally contains several rounds of conversations between a tag and a reader and involves cryptographic [4], [5] operations at both reader and tag sides. Different cryptographic operations adopted by a protocol may generate different timing overhead when executing the protocol.

In general, the time cost of an authentication protocol, as one of the critical concerns in real applications, consists of three parts, namely (1) the time cost for cryptographic operations and looking up a database at the reader side[1], denoted as T_r, (2) the time cost for cryptographic operations at the tag side, denoted as T_t, and (3) the time cost of data exchange between a reader and a tag, denoted as T_c. The speed comparison for cryptographic operations [9] provides a benchmark for the first part of time cost T_r though, currently there is a lack of benchmarks for the latter two parts of time costs T_t and T_c, making it impossible to evaluate the time cost of a protocol amongst difference choices of cryptographic operations.

The difficulty of time cost evaluation for an authentication protocol motivates us to study on the time costs T_t and T_c. In this paper, we conduct comprehensive experiments to test the time costs T_t (running cryptographic operations) and T_c (data exchange between a reader and a tag) based on IAIK UHF tag emulators [10] which follow ISO 18000-6C standard [11], [12] and work as C1G2 passive RFID tags. The cryptographic operations in our study include MD5, SHA-1, SHA-256, SHA-512, AES-128, AES-192, AES-256, and an ultra-lightweight block cipher PRESENT-80 [13], all of which are commonly adopted by authentication protocols [5], [6], [14]. Furthermore, we give a classification for the existing authentication protocols, and summarise a time cost formula for each type of protocols based on the benchmarks of time costs resulting from our experiments.

Our experimental results show that the latter two parts of time costs T_t and T_c are the dominant components of the total time cost for carrying out a protocol, and the first part of time T_r is completely negligible. This is because both T_t and T_c are on the order of ones to hundreds of milliseconds as shown in the next; whereas T_r is below 0.13 millisecond for running cryptographic operations[2] and looking up 2^{20} tags in a database [7]. Furthermore, the results reveal that in most cases over 95% of the total time cost for running a protocol is spent on data exchange, whereas only less than 5% is spent on running cryptographic operations, from which we conclude that in most cases the time cost of data exchange T_c dominates the total time cost for executing a protocol.

Our research results on the benchmarks and formulae provide a fair comparison reference for evaluating the time cost of a protocol under several choices of cryptographic operations

[1]The cryptographic operations and looking up a database are actually carried out by the server to which the reader is connected. Thus the reader side is also referred to as the server side because the reader serves as the system interface to interact with tags.

[2]A single cryptographic operation can be carried out within 2^{-23} second on a PC [7], [9].

in a real application, regardless whether the protocol is an existing one that has been applied to real applications or is to be proposed for certain application scenarios.

The remaining sections are organised as follows. In Section II we present our experimental results for testing the time costs of cryptographic operations and data exchange, following which in Section III we give a classification for the existing authentication protocols and summarise a formula of time cost for each type of protocols. We then in Section IV review the related work about authentication protocols based on the classification before concluding the paper in Section V.

II. Time Costs of Cryptographic Operations and Data Exchange

As aforementioned, the time cost for carrying out an authentication protocol consists of three parts: (1) the time cost at the reader side for running cryptographic operations and looking up a database, denoted as T_r; (2) the time cost at the tag side for running cryptographic operations, denoted as T_t; and (3) the time cost for data exchange between a reader and a tag, denoted as T_c. Data exchange includes sending challenges from a reader to a tag and returning responses from a tag to a reader.

Since time cost T_r is below 0.13 millisecond [7] and is totally negligible as compared with the latter two parts of time costs T_r and T_c which are on the order of ones to hundreds of milliseconds as shown in the next, our experiments will focus on the testing of T_r and T_c which are the dominant components of the total time cost of a protocol. Aiming at this goal, in what follows we conduct four sets of experiments: (1) the first set of experiments will test the time cost for a tag to run the following hashing operations: MD5, SHA-1, SHA-256, and SHA-512; (2) the second set of experiments will test the time cost for a tag to run the following encryption operations: AES-128, AES-192, and AES-256; (3) the third set of experiments will test the time cost for a tag to run PRESENT-80 which is an ultra-lightweight cipher block operation [13]; and (4) the fourth set of experiments will test the time cost of data exchange, i.e, the time cost for a tag to read/write data in its memory banks and the time for a reader to read/write data in the memory banks of a tag.

A. Experimental Setup

The major components in our experimental setup consist of tags and readers.We use two types of tags in our experiments, namely (1) Philips UCODE UHF Gen2 RFID tags (in short as Philips tags henceforth) and (2) IAIK UHF tag emulators [10], a.k.a. DemoTags. DemoTags follow ISO 18000-6C standard [11] and work as C1G2 (class-1 generation-2) passive RFID tags, i.e., the electrical properties such as read/write speeds are almost the same as normal commercial C1G2 tags like Philips tags. A DemoTag is embedded with an ATMega128 micro-controller which is a low-power CMOS 8-bit micro-controller at speed 16 MHz based on the AVR enhanced RISC architecture. A DemoTag (actually the ATmega128 micro-controller) has 128 KB of flash memory, 4 KB of runtime RAM, and 4 KB of non-volatile EEPROM memory. The program of a cryptographic operation is stored in the flash memory, while alterable data (such as EPC, challenge and response) are stored in the EEPROM memory which serves as memory banks of a tag. A DemoTag provides read and write operations, allowing a program stored in the flash memory to read or write *one* 16-bit word of alterable data in each time of read/write operation.

The readers we use in our experiments are CAEN A828 readers [15] equipped with the commands that can read or write several 16-bit words of data in a memory bank of a tag (either a DemoTag or a Philips tag) in each time of read/write operation. Readers are connected to a back-end server which is an IBM T43 laptop PC running Microsoft Windows XP.

The programs running on a DemoTag for cryptographic operations are written in C code on the development platform of Rowley CrossWorks for AVR [16], while the programs running on the server that coordinates a reader to read data from or write data to a DemoTag's EEPROM memory or a Philips tag's memory bank are written with Microsoft Visual C++ 6.0.

TABLE I
Input block size and output size of hashing operations (bytes)

Hashing operation	MD5	SHA-1	SHA-256	SHA-512
n blocks input size	$64n-9$	$64n-9$	$64n-9$	$128n-17$
output size	16	20	32	64

TABLE II
Time cost of hashing operations (milliseconds)

	One block input (bytes)			Two blocks input (bytes)		
	6	30	54	56	90	118
MD5	1.8	1.8	1.8	3.1	3.2	3.1
SHA-1	5.0	5.1	5.0	10.1	10.2	10.1
SHA-256	11.7	11.7	11.6	23.1	23.2	23.1
	6	56	111	112	176	239
SHA-512	41.0	41.1	41.1	81.7	81.8	81.7

B. Experimental Results

The experimental results shown in the next are the average of the results collected from over 500 runs of each set of experiments. Due to the stability of the devices we use (i.e., the readers and tags), the unsuccessful rates of reader to tag read/write operations are below 1%.

(1) In the first set of experiments that test the running time on a DemoTag for the hashing operations of MD5, SHA-1, SHA-256, and SHA-512, we vary the length of input message from one block to two blocks while in each block we take three different input lengths.

Table I shows the input size of blocks and output size for each of the hashing operations. For a given input message, if its length is less than n blocks and greater than $n-1$ blocks (where $n = 1, 2, \dots$), then the hashing operation will cut the input message into $n-1$ blocks and do padding for the last block, after which the hashing is carried out block by block. For any length of input message, the output size (i.e.,

TABLE III
TIME COST OF RUNNING AES (MILLISECONDS)

	AES-128	AES-192	AES-256
Encryption	2.8	3.3	4.3
Decryption	3.1	3.6	4.8

TABLE IV
TIME COST OF LIGHTWEIGHT CRYPTOGRAPHIC OPERATION (MILLISECONDS)

PRESENT-80	1.3

TABLE V
BLOCK SIZE OF AES AND PRESENT-80 (BYTES)

AES-128	AES-192	AES-256	PRESENT-80
16	16	16	8

TABLE VI
TIME COST OF TAG TO TAG READ/WRITE (ONE 16-BIT WORD) OPERATION (MILLISECONDS)

read	0.007
write	16.7

TABLE VII
TIME COST OF READER TO TAG READ/WRITE OPERATION (MILLISECONDS)

words	1	2	3	4	5	6	7
read	46.9	46.9	46.9	47.0	47.1	47.1	47.3
write	63.3	82.9	103.2	124.1	142.3	162.3	183.5
words	8	9	10	11	12	13	14
read	47.1	47.2	47.2	47.2	47.3	47.2	47.2
write	204.0	222.6	243.3	263.6	283.3	303.2	324.5
words	15	16					
read	56.8	57.3					
write	344.6	364.1					

the length of hash value) is the same for each of the hashing operations as shown in Table I.

Table II shows the time cost of running hashing operations with different input lengths. The result tells that the running time of hashing operations increases proportionally with the number of input blocks, but keeps constant if the input length is within the same block size. The time cost of running SHA-1 is about 3 times as that of MD5, while the time cost of running SHA-256 is over two times as that of SHA-1. Given the doubled input block size of SHA-512, the time cost of running SHA-512 is nearly 4 times as that of SHA-256.

(2) In the second set of experiments, we test the time cost for a DemoTag to run encryption operations AES-128, AES-192, and AES-256. With respective 128-bit, 192-bit, and 256-bit key sizes, these three block ciphers have the same 16 bytes of input block length. As shown in Table III, the results tell that the time costs of AES-256 and AES-192 are respectively about 20% and 40% more than that of AES-128. This agrees with the fact that AES-128, AES-192, and AES-256 have respectively 10, 12, and 14 rounds of calculations. Moreover, AES decryption is a bit slower than encryption due to more complex inverse operations.

(3) In the third set of experiments, we test the time cost for a DemoTag to run PRESENT-80 which is an ultra-lightweight block cipher well suitable for extremely constrained environments such as RFID tags and sensor networks [13]. As shown in Table IV, it takes 1.3 ms to run PRESENT-80 with full-round (31 rounds) encryption or decryption operation with 80-bit key size and 8 bytes of input block length. This time cost is nearly 30% less than that of MD5 under one block size of input.

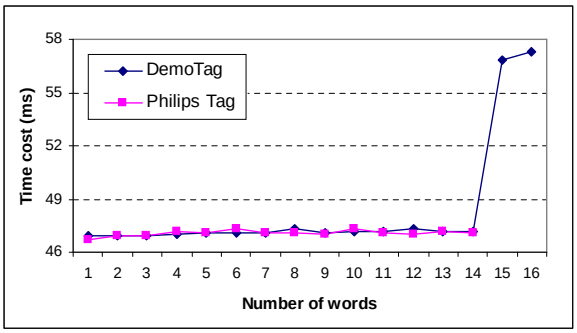
For the block ciphers studied above, namely AES-128, AES-192, AES-256, and PRESENT-80, the input and output are the same length (this length is known as the block size) as shown in Table V. In practice, a protocol that contains cryptographic operations can take any length of input and output. For example, a protocol containing a SHA-1 hashing operation may take 8 bytes as input and 4 out of 20 bytes of hashing value (see Table I) as output; similarly, a protocol containing an AES-128 encryption operation may take only 4 bytes as input and allow padding before running encryption operation, and take 4 out of 16 bytes of block size (see Table V) as output. This implies that the time cost may vary with different lengths of inputs (i.e., challenges from a reader to a tag) and outputs (i.e., responses from a tag to a reader), which will be investigated in detail by the next set of experiments.

(4) In the fourth set of experiments, we investigate (a) the time cost for a tag to read/write data from its memory banks (a.k.a. tag to tag read/write operation) and (b) the time cost for a reader to read/write data in the memory banks of a tag (a.k.a. reader to tag read/write operation).

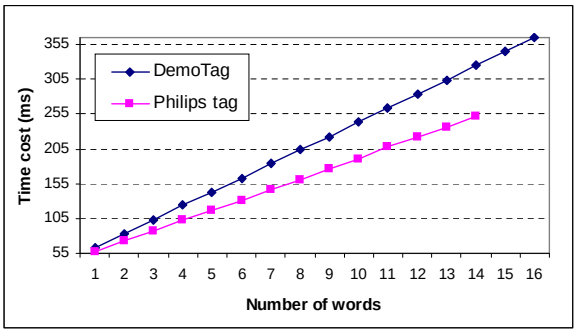
DemoTags provide read and write commands that allow a DemoTag to read from or write to any of its memory banks only *one* 16-bit word of data in each read/write operation. The experimental result as shown in Table VI tells that the time cost is 0.007 ms for a tag to tag read operation and 16.7 ms for a tag to tag write operation. This result reveals that the time cost of such a read operation is completely negligible as compared with that of a write operation.

The CAEN readers we use in our experiments provide read and write commands that allow a reader to read from or write to a memory bank of a tag (either a DemoTag or a Philips tag) n 16-bit words of data (where $n = 1, 2, \dots$) in each of its read/write operation. In the experiments, we let a CAEN reader read the user memory bank of a DemoTag and a Philips tag. There are 16 words of memory space in the user memory bank of a DemoTag, and 14 words of memory space in the user memory bank a Philips tag. We increase the number of words accessed by a reader one by one.

The time costs of read/write operation on DemoTag's user memory bank are shown in Table VII, which tells that (a) the time cost of a read operation keeps stable at around 47 ms for reading of one to 14 words, and jumps to 56.8 ms at reading of 15 words and tends to increases linearly but slightly with an increment of around 0.5 ms for reading of 16 words; and (b) writing a single word takes nearly 63 ms which is about



(a) Read operation



(b) Write operation

Fig. 1. Time cost of reader to tag read/write operations

four times of a tag to tag write operation, and the time cost increases linearly with the number of words at an increment of about 20 ms.

Figure 1 gives a comparison of the time costs of read/write operations for both types of tags, in which Figure 1(a) shows that the time costs of read operation are the same (i.e., at around 47 ms) for both tags in reading the first 14 words; while Figures 1(b) shows that the time cost of write operation on Philips tags, which increases linearly with the number of words at an increment of about 15 ms, is slightly less than that on DemoTags.

III. Protocol Classification and Time Cost Evaluation

In what follows, we give a classification for the existing protocols and summarise a formula of time cost evaluation for each type of authentication protocols. With the above benchmarks of time costs, we give several examples of time cost evaluation for some protocols based on the formulae. From the examples and the benchmarks, we conclude that the time cost for carrying out a protocol is mainly determined by the time cost of data exchange which is the dominant component of the total time cost.

A. Classification of Reader/Tag Authentication Protocols

A typical reader/tag authentication protocol works by two phases, namely, a tag identification phase followed by an authentication phase [6], [17], [18]. There are many reader/tag

1. Tag $\longleftarrow$ Reader: c_1
   if (c_1 is correct) **then** accept the reader
   else reject the reader

(a) RAO protocols

1. Reader $\longrightarrow$ Tag: c_1
2. Reader $\longleftarrow$ Tag: $R(c_1)$
3. Server $\longleftarrow$ Reader: $R(c_1)$
   if ($R(c_1)$ is correct) **then** accept the tag
   else reject the tag

(b) TAO protocols

1. Reader $\longrightarrow$ Tag: c_1
2. Reader $\longleftarrow$ Tag: $R(c_1)$
3. Server $\longleftarrow$ Reader: $R(c_1)$
   if ($R(c_1)$ is correct) **then** accept the tag
   else reject the tag and abort
4. Tag $\longleftarrow$ Reader: c_2
   if (c_2 is correct) **then** accept the reader
   else reject the reader

(c) TRA protocols

1. Tag $\longleftarrow$ Reader: c_1
   if (c_1 is correct) **then** accept the reader
   else reject the reader and abort
2. Reader $\longleftarrow$ Tag: $R(c_1)$
3. Server $\longleftarrow$ Reader: $R(c_1)$
   if ($R(c_1)$ is correct) **then** accept the tag
   else reject the tag

(d) RTA protocols

Fig. 2. Illustration of simplified four types of authentication protocols

authentication protocols with different rounds of conversations in the authentication phase though, the tag identification phase is the same amongst all. If we ignore the tag identification phase which includes the anti-collision procedure and inventory command, the existing reader/tag authentication protocols can be classified into *four* types based on the conversations (between a reader and a tag) in their authentication phases. They can be simplified as illustrated in Figure 2. The first two types of protocols are one-way authentications, i.e., either a tag authenticates itself to a reader or a reader authenticates itself to a tag; whereas the latter two types of protocols are mutual authentications by which tags and readers authenticate to each other mutually. They are briefly described as follows.

(1) The first type is reader authentication protocols under which only readers authenticate themselves to tags, referred to as RAO (reader authentication only) as shown in Figure 2(a). Challenge c_1 sent by the reader is a function (such as a hashing operation) of key k held by the tag. The tag runs the same function with key k as the input and compares the result with the received challenge c_1. If c_1 is equal to the result, i.e., c_1 is

correct as illustrated in the figure, then it accepts the reader; otherwise it rejects the reader. With hash-locking protocol [17] as a representative, this type of protocols prevent unauthorised readers from accessing tags.

(2) The second type is tag authentication protocols under which only tags authenticate themselves to readers, referred to as TAO (tag authentication only) as shown in Figure 2(b). Upon receiving challenge c_1 sent by the reader, the tag (of which the legitimacy is unknown) returns response $R(c_1)$ which is a function of challenge c_1. The reader verifies the correctness of $R(c_1)$ on the back-end server, and accepts the tag if correct or rejects the tag otherwise. This type of protocols allow readers to prevent unauthorised tags from being accepted. Representatives of this type of protocols include Tsudik's YA-TRAP [4] and five protocols of challenge-response proposed by Vajda and Buttyán [19].

(3) The third type of protocols such as the protocol of Avoine *et al.* [7], the protocol of Yang *et al.* [20] and the protocol of Molner and Wagner [8], referred to as TRA (tag-then-reader authentication) in Figure 2(c), allow tags to authenticate themselves to readers first followed by readers authenticating themselves to tags. TRA protocols work as a combination of TAO and RAO. The reader first sends a random number c_1 as the challenge and receives response $R(c_1)$, and verifies its correctness on the back-end server. If correct, then it accepts the tag and proceeds to authenticating itself to the tag by sending another challenge c_2; otherwise, it rejects the tag and aborts the authentication process. Once the tag passes the authentication to the reader, it verifies challenge c_2. If it is correct, then the tag accepts the reader; otherwise, it rejects the reader.

(4) The fourth type of protocols such as the protocol of Alomari *et al.* [6] and M^{2}AP of Peris-Lopez *et al.* [14], referred to as RTA (reader-then-tag authentication) in Figure 2(d), allow readers to authenticate themselves to tags first and then tags authenticate themselves to readers. Similar to TRA protocols, RTA protocols work as a combination of RAO and TAO. The difference is, under TRA protocols, the tag responds challenge c_1 without verifying its validity; whereas under RTA protocols, the tag first verifies challenge c_1 before responding with response $R(c_1)$. If c_1 is correct, then the tag accepts the reader; otherwise, it rejects the reader and aborts the authentication. Once the reader passes the authentication to the tag, it starts to authenticate the tag by verifying the correctness of response $R(c_1)$ on the back-end server. If correct, then it accepts the tag; otherwise, it rejects the tag.

B. Formulae of Time Cost Evaluation

We are going to use some notations as shown in Table VIII for the formulae of time cost evaluation. As mentioned in the previous section, in general the time cost of the protocol comprises of three parts as described by formula $T = T_t + T_r + T_c$. We give a further explanation for terms T_t and T_c in what follows.

(1) In our experiments, T_t the time cost of running cryptographic operations at the tag side is actually comprised of two parts, namely (a) the time cost of running the operations, and (b) the time cost for a tag to read the input of a cryptographic operation from it memory bank, denoted as $m \cdot r_t$ where m is the number of words it reads. An input is normally a challenge (or part of a challenge) sent from a reader. From Table VI we can see that, as compared with other operations, the time cost of tag to tag read operation is below 0.01 millisecond and is completely negligible. Normally the input is far less than 100 words, and T_t can be approximated by the first part of time cost.

TABLE VIII
NOTATIONS

T	time cost of carrying out a protocol
T_r	time cost of a cryptographic operation and looking up a database at the reader side
T_t	time cost of a cryptographic operation at the tag side
T_c	time cost of data exchange between a reader and a tag
r_t	time cost of a tag to tag read operation for reading one word
w_t	time cost of a tag to tag write operation for writing one word
r_n	time cost of a reader to tag reader operation for reading n words
w_n	time cost of a reader to tag write operation for writing n words
n	reader accessing n words of data from tag's memory bank

(2) The time cost of data exchange T_c contains two parts, namely (a) the time cost that a reader sends challenges, which is equivalent to reader to tag write operations, and (b) the time cost that a tag returns responses. For C1G2 passive tags such as DemoTags or Philips tags, due to their passiveness in data exchange, returning a response is actually carried out by two steps, namely (i) a tag first writes its result of a cryptographic operation to one of its memory banks, and (ii) a reader reads the result from tag's memory bank. Thus returning a response is equivalent to a tag to tag write operation followed by a reader to tag read operation.

Considering that T_r is below 0.13 millisecond [7] and is completely negligible as compared with T_t and T_c which are on the order of ones to hundreds milliseconds as shown in the previous section, the formula of the total time cost can be simplified as follows:

$$T = T_t + T_c$$

With this simplified formula of time cost evaluation, we give the detailed formula for each type of authentication protocols in what follows based on the protocol classification.

(1) RAO protocols. This type of protocols contains (a) one round of reader to tag write operation, which is corresponding to sending challenge c_1 to the tag as shown by step 1 in Figure 2(a), and (b) a cryptographic operation at the tag side. Normally challenge c_1 is $f(x)$ the result of a cryptographic operation with input x, in which x is the secret key k held by the tag[3] [8], [14], [17]. To authenticate the legitimacy of a reader, a tag runs the same cryptographic operation with input x, and compares the result with $f(x)$ it receives as challenge c_1. The formula of time cost is given as follows:

$$T = w_n + T_t \tag{1}$$

[3]It is possible that x is a concatenation of a random number r with key k, and correspondingly challenge c_1 is the concatenation of r and $f(x)$.

in which n is the number of words in challenge c_1.

(2) TAO protocols. This type of protocols contains (a) one round of reader to tag write operation, which is corresponding to sending challenge c_1 as shown by step 1 in Figure 2(b), (b) a cryptographic operation at the tag side, which is corresponding to tag's calculating response $R(c_1)$, and (c) one round of tag to tag write operation followed by one round of reader to tag read operation, which is corresponding to returning response $R(c_1)$ as shown by step 2 in Figure 2(b). Here the challenge c_1 could be a random number, and the response $R(c_1)$ is the cryptographic result $f(c_1, k)$ with inputs c_1 and the secret key k that the tag holds [4], [5], [19]. The formula is given as follows:

$$T = w_n + T_t + m \cdot w_t + r_m \tag{2}$$

in which n is the number of words in challenge c_1 and m the number of words in the cryptographic result $f(c_1, k)$.

(3) TRA protocols. This type of protocols is a combination of the previous two types of protocols. It contains (a) two rounds of reader to tag write operation, which are corresponding to reader's sending challenges c_1 and c_2 as shown by steps 1 and 2 in Figure 2(c), (b) two cryptographic operations at the tag side, which are corresponding to tag's calculating $R(c_1)$ and verifying c_2 as shown by steps 2 and 4 in Figure 2(c), and (c) one round of tag to tag write operation followed by one round of reader to tag read operation, which is corresponding to returning response $R(c_1)$ as shown by step 2 in Figure 2(c). In the process of data exchange, challenge c_1 could be a random number, whereas challenge c_2 is the result of a cryptographic operation $f(x)$ with input x. The formula is given as follows:

$$T = w_{n_1} + T_t + m \cdot w_t + r_m + w_{n_2} + T'_t \tag{3}$$

in which T_t is the time cost of generating response $R(c_1)$, T'_t the time cost of running $f(x)$, n_1 the number of words in challenge c_1, m the number of words in response $R(c_1)$, and n_2 the number of words in challenge c_2.

(4) RTA protocols. As another combination of RTO and TRO protocols, this type of protocols contains (a) one round of reader to tag write operation, which is corresponding to sending challenge c_1 as shown by step 1 in Figure 2(d), (b) two cryptographic operations at the tag side, which are corresponding to tag's verifying challenge c_1 and calculating response $R(c_1)$ as shown by steps 1 and 2 in Figure 2(d), and (c) one round of tag to tag write operation followed by one round of reader to tag read operations, which is corresponding to returning response $R(c_1)$ as shown by step 2 in Figure 2(d). In the protocol, challenge c_1 is the result of a cryptographic operation $f(x)$ with inputs x. The formula is given as follows:

$$T = w_n + T_t + T'_t + m \cdot w_t + r_m \tag{4}$$

in which T_t is the time cost of running a cryptographic operation with input x, T'_t the time cost of generating response $R(c_1)$ which is the result of another cryptographic operation with input c_1 and secret key k, n the number of words in challenge c_1, and m the number of words in response $R(c_1)$.

TABLE IX
TIME COST OF DATA EXCHANGE T_c WITH DIFFERENT LENGTHS OF CHALLENGE/RESPONSE (MS)

length	RAO	TRO	TRA	RTA
1 word	63.3	126.9	190.2	126.9
4 words	124.1	237.9	362.0	237.9
8 words	204.0	384.7	588.7	384.7
10 words	243.3	457.5	700.8	457.5
12 words	283.3	531.0	814.3	531.0

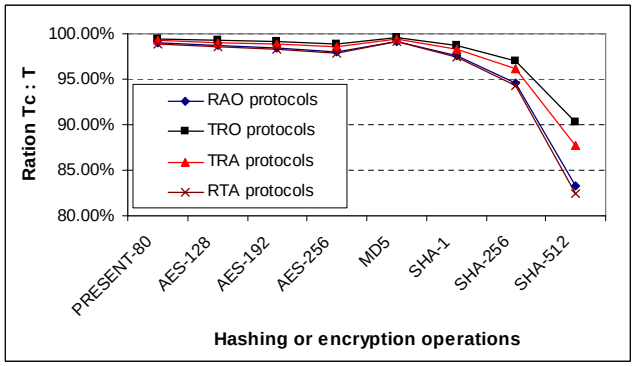


Fig. 3. Ratio of time cost of data exchange vs. total time cost of carrying out a protocol

C. Example of Time Cost Evaluation

If we just consider the time cost of data exchange T_c and assume the same length of input and output (i.e., challenge and response equivalently), then we can extract T_c from Formulae (1) to (4) as shown below:

$$\begin{aligned} \text{RAO}: \ T_c &= w_n \\ \text{TAO}: \ T_c &= w_n + r_n + n \cdot w_t \\ \text{TRA}: \ T_c &= 2w_n + r_n + n \cdot w_t \\ \text{RTA}: \ T_c &= w_n + r_n + n \cdot w_t \end{aligned}$$

from which we get a set of values of T_c with various length of input or output as shown in Table IX. The data in the table shows that the time cost of data exchange T_c increases with the length of input or output. Specifically, the second line of Table IX, in which the input or output size is one word, shows the lower bound of the time cost of data exchange time T_c.

If we let the input and output take the full block size for AES and PRESENT-80 encryption operations and MD5 hashing operation, i.e., 8 words for AES and MD5, and 4 words for PRESENT-80, and take 8 words as input and 8 words as part of output size for SHA-1, SHA-256, and SHA-512 hashing operations, then by looking up Tables II to IV, based on Formulae (1) to (4) we get the ratio of the time cost of data exchange T_c vs. the total time cost of a protocol T as shows in Figure 3. From the figure, we can draw the following conclusions: (a) in most cases over 95% of the total time cost for running a protocol are spent on data exchange, whereas only less than 5% are spent on running cryptographic operations. Even for the worst case of SHA-512, the time

cost of data exchange takes over 80% of the total time cost, (b) the ratio drops with the increase of time cost for running a cryptographic operation, i.e., the ratio drops from PRESENT-80 to AES-256 and from MD5 to SHA-512, and (c) with the data shown in Table IX, we can further conclude that the ratio will increases with the length of input or output. Therefore, the above results reveal that in most cases the time cost of data exchange T_c completely dominates the total time cost for running a protocol.

In real applications, the cryptographic operations are normally implemented with circuits of logic gates and hence the time cost of running such an operation is estimated faster (say 2 to 10 times faster) than our software implementation[4]. Meanwhile, it is possible to increase the speed of read/write operations at most several times based on ISO standard for C1G2 tags. However, these would further increase the percentage of data exchange time out of the total time cost for executing a protocol. Therefore, our study reveals that the key to reduce the total time cost of running a protocol is to minimise the length of inputs and outputs and to cut off the rounds of data exchange.

IV. Related Work

There are many papers that theoretically discuss the implementation of authentication protocols or calculate the number of logic gates and time cost for implementation. For example, Avoine *et al.* proposed a solution which greatly reduces the time of looking up a database to authenticate the legitimacy of an RFID tag, and evaluated the number of logic gates for the implementation of cryptographic operations. Bogdanov *et al.* [21] discussed compact hashing operations on RFID tags and estimated the number of logic gates for implementation. They also discussed the implementation of PRESENT-80 as an ultra-lightweight block cipher [13]. However, there are only a few papers addressing protocols or operations that have been implemented on real C1G2 tags or emulators, mainly due to that currently there are only a few of tag emulators that are programmable and allow to run different cryptographic operations. A remarkable work of implementation is done by Fu's research group [22] with WISP tags [23] which is another tag emulator besides DemoTags we are using. They implemented RC5 [24] on WISP tags based on which they gave measurements of computation and symmetric cryptography. Another work by Fu's group implemented on WISP tags is reported by [25] in which they proposed a framework and gave a measurement for energy management in CRFIDs (computational RFIDs).

As for reader/tag authentication protocols, there are many of them that have been proposed for reader and tag authentications. Based on the application environments, some protocols only fulfill one-way authentications [4], [5], [17], i.e., either tags authenticate to readers or the reverse, while some other protocols fulfill mutual authentications (see, e.g., [6], [7], [14], [20], [26]), i.e., both readers and tags authenticate to each other mutually, leading to different time costs for carrying out the protocols.

To prevent unauthorised readers from accessing legitimate tags, Weis *et al.* proposed the hash-locking protocol [17] which enables a tag to authenticate the legitimacy of a reader first before allowing the reader to access the tag's memory. Under this protocol, if a challenge sent from a reader is a function of the secret key held by a tag, then this reader is accepted by the tag and allowed to access the tag's memory; otherwise, it is rejected by the tag. In real applications, this type of protocols only takes a very small portion of all reader/tag authentication protocols.

In most applications, readers are guaranteed legitimate but tags could be counterfeit. Thus some protocols have been proposed to prevent unauthorised tags from being accepted by readers. The major difference amongst these protocols is how a response is generated. One of the representatives of such protocols is Tsudik's YA-TRAP [4]. With YA-TRAP protocol, a response from a tag is a hashing result of the secret key held by the tag and the challenge sent from a reader. Upon receiving the response, the reader transfers it to the back-end server together with the challenge. The server runs the same hashing operation with the challenge and each secret key in its database, and compares the results so as to verify the legitimacy of the tag. Feldhofer *et al.* [5] implemented such a protocol under which a response is generated with AES algorithm. Vajda and Buttyán [19] proposed five protocols of challenge-response with different ways of response generating for a tag to authenticate itself to a reader. Henrici and Müller [26] proposed a protocol to authenticate tags which at the same time can preserve the privacy of tags' locations.

In some circumstances tags may suffer from attacks coming from counterfeit readers, and it is necessary for a reader and a tag to mutually authenticate to each other. Mutual authentications are combinations of the above one-way authentications with two possible orders, i.e., either a reader first authenticates itself to a tag [6], [14] or the reverse [7], [8], [20], [27]. Alomair *et al.* [6] proposed a mutual authentication protocol which they claimed unconditionally secure. The protocol only used primitive operations (i.e., bitwise XOR operations, modular multiplication and addition) on tags so as to achieve low-cost computations. Peris-Lopez *et al.* proposed M^2AP protocol [14] which contains only two rounds of conversation between a reader and a tag and uses primitive operations. They claimed it as the minimalist mutual-authentication protocol for low-cost RFID tags, and studied the security property that Yang *et al.* analysed for their protocol proposed in [20]. Other properties such as time complexity and privacy-preserving of lightweight protocols have been studied in [7], [8], [27].

V. Conclusions

In conclusion, we have made the following contributions in the paper: (1) we have identified three major components of the time cost for carrying out an authentication protocol,

[4]There is a claim from some providers of tag emulators that the relative time cost of software implementation is comparable to hardware implementation though, so far there is not any report claiming that software implementation is two or more times faster than hardware implementation.

(2) we have conducted comprehensive experiments to test the time cost for running cryptographic operations and the time cost for data exchange between a reader and a tag based on IAIK UHF tag emulators and CAEN readers, (3) with the benchmarks of time costs coming out from our experiments, we have given a classification for the existing authentication protocols and summarised a time cost formula for each type of the protocols, and (4) from the experimental results and the formulae, we have concluded that in most cases the time cost of data exchange completely dominates the total time cost of running a protocol. Our research results can serve as a fair comparison reference for evaluating the time cost of an authentication protocol.

For the next stage of study, it would be interesting to extend our experiments to asymmetric cryptographic systems such as ECC and RSA, and to carry out the experiments on WISP tags.

ACKNOWLEDGEMENT

This work is partly supported by Singapore A*STAR SERC Grant No. 082-101-0022.

REFERENCES

[1] Roberts, C. M., "Radio frequency identification (RFID)," *Computers and Security*, vol. 25, no. 1, pp. 18–26, 2006.

[2] Angeles, R., "RFID technologies: supply-chain applications and implementation issues," *Information Systems Management*, vol. 22, no. 1, pp. 51–65, 2005.

[3] Melski, A., J. Müller, A. Zeier, and M. Schumann, "Improving supply chain visibility through RFID data," in *Proceedings of the the IEEE 24th International Conference on Data Engineering Workshop (ICDEW'08)*, Cancun, Mexico, Apr. 7–12, 2008, pp. 102–103.

[4] Tsudik, G., "YA-TRAP: Yet another trivial RFID authentication protocol," in *Proceedings of the Fourth IEEE Annual International Conference on Pervasive Computing and Communications Workshops (PerComW 2006)*, Pissa, Italy, Mar. 13–17, 2006, pp. 643–646.

[5] Feldhofer, M., S. Dominikus, and J. Wolkerstorfer, "Strong authentication for RFID systems using the AES algorithm," in *Proceedings of the Sixth International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004)*, Cambridge, MA, USA, Aug. 11–13, 2004, pp. 357–370.

[6] Alomair, B., L. Lazos, and R. Poovendran, "Towards securing low-cost RFID systems: an unconditionally secure approach," in *Proceedings of the 2010 Workshop on RFID Security (RFIDsec'10 Asia)*, Singapore, Feb. 22–23, 2010, pp. 1–17.

[7] Avoine, G., E. Dysli, and P. Oechslin, "Reducing time complexity in RFID systems," in *Proceedings of the 12th Annual Workshop on Selected Areas in Cryptography (SAC'05)*, Kingston, ON, Canada, Aug. 11–12, 2005, pp. 291–306.

[8] Molner, D. and D. Wagner, "Privacy and security in library RFID: Issues, practices, and architectures," in *Proceedings of the 11th ACM conference on Computer and communications security (CCS'04)*, Washington DC, USA, Oct. 25–29, 2004, pp. 210–219.

[9] Crypto++ Benchmarks, available at `http://www.cryptopp.com/benchmarks.html`.

[10] DemoTag, `http://www.iaik.tugraz.at/content/research/rfid/tag_emulators/`.

[11] ISO 18000-6C, `http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=34117`.

[12] EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz–960 MHz Version 1.2.0, `http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_2_0-standard-20080511.pdf`.

[13] Bogdanov, A., L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," in *Proceedings of the 9th International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2007)*, Vienna, Austria, Oct. 10–13, 2007, pp. 450–466.

[14] Peris-Lopez, P., J. C. Hernández-Castro, J. M. Estévez-Tapiador, and A. Ribagorda, "M^2AP: A minimalist mutual-authentication protocol for low-cost RFID tags," in *Proceedings of the Third International Conference on Ubiquitous Intelligence and Computing (UIC06)*, Wuhan, China, Sep. 3–6, 2006, pp. 912–923.

[15] CEAN RFID, *CAENRFIDLib: Ansi C functions Library—Technical information manual*, `http://www.caen.it/rfid/index.php`.

[16] Rowley CrossWorks, `http://www.rowley.co.uk/avr/index.htm`.

[17] Weis, S. A., S. E. Sarma, R. L. Rivest, and D. W. Engels, "Security and privacy aspects of low-cost radio frequency identification systems," in *Proceedings of the First International Conference on Security in Pervasive Computing (SPC 2003)*, Boppard, Germany, Mar. 12–14, 2003, pp. 201–212.

[18] Lai, Y.-C. and C.-C. Lin, "Two blocking algorithms on adaptive binary splitting: Single and pair resolutions for RFID tag identification," *IEEE/ACM Transactions on Networking*, vol. 17, no. 3, pp. 962–975, 2009.

[19] Vajda, I. and L. Buttyán, "Leightweight authentication protocols for low-cost RFID tags," in *Proceedings of the Fifth International Conference on Ubiquitous Computing (UbiComp 2003)*, Seattle, WA, USA, Oct. 12–15, 2003.

[20] Yang, J., J. Park, H. Lee, K. Ren, and K. Kim, "Mutual authentication protocol for low-cost RFID," in *Proceedings of the Workshop on RFID and Lightweight Crypto*, Graz, Austria, Jul. 14–15, 2005, pp. 17–24.

[21] Bogdanov, A., G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, and Y. Seurin, "Hash functions and RFID tags: Mind the gap," in *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2008)*, Washington, DC, USA, Aug. 10–13, 2008, pp. 283–299.

[22] Chae, H.-J., D. J. Yeager, J. R. Smith, and K. Fu, "Maximalist cryptography and computation on the WISP UHF RFID tag," in *The Conference on RFID Security 2007 (RFIDSec 07)*, Malaga, Spain, Jul. 11–13, 2007.

[23] Smith, J. R., A. P. Sample, P. S. Powledge, S. Roy, and A. Mamishev, "A wirelessly-powered platform for sensing and computation," in *Proceedings of the Eighth International Conference on Ubiquitous Computing (Ubicomp 2006)*, Orange County, CA, USA, Sep. 17–21, 2006, pp. 495–506.

[24] Rivest, R. L., "The RC5 encryption algorithm," in *Proceedings of the Second International Workshop on Fast Software Encryption (FSE94)*, Leuven, Belgium, Dec. 14–16, 1994, pp. 86–96.

[25] Ransford, B., S. Clark, M. Salajegheh, and K. Fu, "Getting things done on computational RFIDs with energy-aware checkpointing and voltage-aware scheduling," in *USENIX Workshop on Power Aware Computing and Systems (HotPower 2008)*, San Diego, CA, USA, Dec. 7, 2008.

[26] Henrici, D. and P. Müller, "Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers," in *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PerComW 2004)*, Orlando, FL, USA, Mar. 14–17, 2004, pp. 149–153.

[27] Yu, S., K. Ren, and W. Lou, "Privacy-preserving lightweight authentication protocol for low-cost RFID tags," in *Proceedings of the IEEE Military Communication Conference (MILCOM 2007)*, Orlando, FL, USA, Oct. 29–31, 2007, pp. 1–7.