Performance Evaluation of Brands' Digital Cash System in TCP/IP Environment

Pock-Chueng Teo
Revol Technologies;
Singapore
lawrenceteo@yahoo.com

Ahmad Zuri Sha'ameri
Universiti Teknologi
Malaysia

Mohsen Ashourian
Islamic Azad University of
Iran-Majlesi Branch

Abstract — Brands' digital cash system is a single term system that builds on two concepts, Schnorr digital signature and the representation problem in group of prime order. The Brands' scheme is almost as efficient as fully traceable offline systems. In this paper, we evaluate the performance of Brands' system in TCP/IP Environment and report the results in various conditions.

1. INTRODUCTION

The advent of electronics changes the face of the world. The advantage of electronics can be defined in three categories – computation, communication and storage, with the features of fast, accurate and persistent. Consequently, electronics found its applications in a wide variety of domains. In the endless list of application domains, commerce is not an exception. Electronic commerce is a contemporary term denoting the commerce activities conducted using electronic mechanism, either directly or indirectly, in front of or behind the scene. Computation, communication and storage are the three elementary components in electronic commerce that work in a highly interactive manner to fully exploit the advantage that could not be obtained in the traditional commerce. Electronic payment as the final stage in electronic commerce plays an important role. Without proper and efficient electronic payment mechanism, electronic commerce cannot fully demonstrate its real potential. Digital cash is a class of electronic payment that emphasizes security, convenience, and anonymity.

2. BRANDS' DIGITAL CASH SCHEME

The Brands' digital cash scheme is an offline anonymous digital cash scheme developed based on an underlying number theory hard problem known as the 'representation problem' [1]. The representation problem is believed to be equivalent in computational difficulty to computing discrete logarithms, rather than RSA root. Hence, the mathematical breakthrough in factoring method does not reduce the security of Brands' digital cash system.

Brands' digital cash scheme involves three entities and consists of four protocols. A bank plays the role of an authorized digital cash issuer. A user withdraws digital cash from the bank and pays the digital cash to a merchant in exchange for goods and services. The merchant finally deposits the digital cash back to the bank. The Brands' digital cash transaction is therefore the circulation of the Brands' digital cash token between the three entities. This paper uses the terms merchant and shop interchangeably to denote the party who sells goods or services to the user.

The Brands' scheme is an untraceable offline digital cash scheme. Due to the nature of the two features – untraceable and offline, the integration into a single scheme is not trivial. The Brands' scheme applies subtle techniques to simultaneously achieve the two primary aspects of security and anonymity. The bank digitally signs on digital cash token in order to make the token valid for transaction. The digital signature scheme used by the bank is sufficiently strong to resist the possibility of signature forgery by attackers. To protect the anonymity of the user, Brands' scheme allows the user to execute restrictive blinding on the digital cash token to be signed. The restrictive blinding ensures that the user can only change the external appearance of the token while the internal token structure embedded with the user's identity information shall remain intact. This implies that the user is not able to destroy the internal token structure, which aids owner tracing in the case of double spending, through the blinding process. The restrictive blinding prevents the bank from being able to match the withdrawn and deposited digital cash tokens and thus protecting the user anonymity.

To use the digital cash token in the payment protocol, the user needs to prove his ownership over the token. The ownership is proven through the knowledge of representation on the token. Under the assumptions of Discrete Logarithm and Diffie-Hellman, no one except the legal token owner could provide such a proof of ownership. Additionally, the user would release partial information during the proof of ownership. A single piece of such partial information does not lead to the revelation of the user identity; however two pieces do. The bank is able to trace to the double spender through this mechanism. In other words, the Brands' scheme provides a means to expose the double spender's identity after the fact.

3. IMPLEMENTATION OF BRANDS' SYSTEM

The Brands' scheme consists of 4 different protocols – namely Setup, Withdrawal, Payment and Deposit. Account Setup needs to be done only once when the user creates an account at the particular bank. Thereafter, the user needs to carry out the remaining three protocols for future digital cash transaction. A complete digital cash

life cycle covers Withdrawal, Payment and Deposit protocols.

In order to empirically assess the performance of the Brands' system, we setup a simulation environment in which two remote hosts communicating using TCP/IP networking could execute a series of Brands' digital cash transactions. Detail of the implementation can be found in [2]. The two remote hosts are geographically separated to provide a modeling environment, which is closer to reality. In addition, the simulation identified and included a few crucial control variables, which could unveil the overall performance in broader aspects.

Computation and communication are the two major mechanisms in digital cash system. Therefore, the simulation measures the performance in term of time interval for computation and communication. The computation and communication time intervals are measured in millisecond unit. The time interval for computation is rather straightforward. The way computer handles computation is in synchronous mode – computation operations are executed serially and one starts immediately after another ends. Therefore the time interval measurement for computation is accurately describing how long the computation takes to accomplish. To minimize the interrupt from the operating system scheduler, we provide a working system that has minimum executing process. We can assume that most interrupts come from the operating system, instead of other applications.

However, things become more complicated for the case of communication. Due to the nature of TCP implementation, the communication time is used to describe the time interval for communication to accomplish the corresponding request. The timer keeps track of the time interval for the sending or receiving request from the beginning to the end. In most of the cases, the sending request does not block (the sending request function returns immediately) because the TCP entity will take over the responsibility to queue and dispatch the sending content. The sending request blocks only if the TCP entity has not enough buffer to process the sending request. However, it is quite rare for sending request to block in today's technology as the TCP entity always has sufficient buffer. It is the decision of the TCP entity to send or accumulate for larger communication block, for performance reason. On the other hand, it is more likely for the receiving request blocks until the communication content is ready. When the receiving request ends its blocking, it implies that the TCP entity has handed over the desired receiving content to the application entity.

In short, the communication time in the context of this paper does not represent the traverse time for the data packet to reach the destination from the source. Instead, the communication time is normally equivalent to the waiting time to accomplish the communication request plus receiving time to obtain all the data from the source. The word 'normally' is used in the previous sentence to describe there is an uncertainty in the statement as TCP entity is really a dynamic element that is out of the application entity's control. This is the justification to the semantics of communication time used in this paper and it will be used throughout this paper implicitly.

4. Performance Evaluation Factors

To perform a more complete evaluation on the developed Brands' digital cash model, various control variables of interest are taken into consideration. The identified control variables include parameter bit length, process and thread priority in multitasking operating system and computer system configuration.

Parameter Bit Length

Parameter bit length is an important indicator to the security level of digital cash scheme, as well as other cryptographic algorithms. Brands' scheme relies on the bit lengths of parameter p and q for the security level. The bit lengths of the remaining parameters, found in the protocols of Brands' scheme, are functions of the bit lengths of p and q. It implies that the determination of the bit lengths of the parameter p and q has setup a framework for the bit lengths of the other parameters. In the remaining part, the parameter bit length implicitly refers to the bit length for parameter p. q is implicitly assumed as half of the size of parameter p.

As the same in the cryptography convention, the longer bit length of a parameter, the safer the system is. Nevertheless, the trade off is the computation time. There is theoretically no totally unbreakable hard problem in the current world of public key cryptography, so as the Brands' scheme. The implication stems from the fact that the Brands' scheme is simply a collection of protocols based on the representation problem, a hard problem in number theory. Finding a balance point of compromise between the security and feasible computing time is critical. Schnorr [3] suggested that the bit length for p is 512 bits and the bit length for q is 140 bits. However, the computing technology over the years has significant advancement since Schnorr's recommendation in 1989. DSA (Digital Signature Algorithm) uses p with the bit length between 512 and 1024, as well as q with the bit length of 160 [4].

To maintain high degree of security, this paper uses a standard bit lengths of 1024 and 512 bits, respectively for p and q. On the other hand, the simulation empirically tested a range of bit lengths, centred on the standard bit lengths. The purpose is to uncover the performance of the developed digital cash system in relation to a wide variety of bit length settings.

Process and Thread Priority

Modern advanced operating systems are mostly multitasking systems. Windows, UNIX, Solaris and Linux are the examples of multitasking operating systems. Multitasking operating system allows multiple executables to share the resources of computer, especially

the microprocessor, simultaneously. In fact, a microprocessor can only process a single task at any particular time. For single processor system, multiple executables take turns to use the processor in a very short duration and frequent manner, thus create an illusion of simultaneous executions of multiple processes to the computer users. The operating systems are responsible to schedule the timetable for executables to use the microprocessor.

Threads compete with each other to acquire the execution right on the processor. Operating system scheduler determines which thread to gain execution control and when the thread runs, by having a set of criteria. Priority is the one of the essential elements in the criteria. Thread of higher priority has privilege over thread of lower priority. The priority determines the way that an executable to be scheduled for microprocessor consumption. The higher priority a thread has, the higher possibility that it obtains the resource for execution. However, in the operating system, many other system processes, e.g. user interface process, I/O processes and scheduler process, are running at the same time, racing to get the computing resource for execution, in order to keep the system up and running. Over competing for resources with the system processes causes the system to be less responsive and sometimes fails. The setting of thread priority should be fair and reasonable. Having all threads to possess high priority status does not make any sense. High priority should be given to time-critical mission threads for a period as short as possible. After completing the time-critical mission, the high priority status should be disabled. Thread starvation happens when high priority thread is occupying the execution control on processor for a long time until the other threads do not obtain sufficient resources to execute.

Proper thread priority could help to increase the performance of the digital cash system while maintaining the optimal working state of the operating system. The simulation investigates the effect brought by the different levels of thread priority to the performance of the digital cash system as well as the operating system. A numbering convention has been established in this paper (listed in Table 1) to denote different priorities throughout the section of performance evaluation. The following priority levels are only a subset from the Windows thread priority family. In this paper, we focus on the impact and influence of thread priority to the performance of digital cash system and will not include process priority.

Table 1. The Numbering Convention for Thread Priority

Thread Priority	Number
THREAD_PRIORITY_IDLE	0
THREAD_PRIORITY_LOWEST	1
THREAD_PRIORITY_BELOW_NORMAL	2
THREAD_PRIORITY_NORMAL	3
THREAD_PRIORITY_ABOVE_NORMAL	4
THREAD_PRIORITY_HIGHEST	5
THREAD_PRIORITY_TIME_CRITICAL	6

Computer System Configuration

Different configurations of computer system, such as the microprocessor, RAM, hard disk, network card and operating system have different performances. The simulation reviews the issues to discover what impact the computer system configuration brings to the digital cash transaction. The server is hosted on a Pentium 4 (1.5 GHz) personal computer with Windows 2000 (Service Pack 2). The Table 2 gives the detailed system configuration.

Table 2. The Server System Configuration

CPU	Intel Pentium 4
Clock Speed	1.5 GHz
RAM	512 MB
Hard Disk	10 GB
Operating System	Microsoft Windows 2000 Professional (Service Pack 2)

The computation performance has direct relationship with the system computer configuration and the operating system. Unlike the remote clients, the system configuration of the server is constant throughout the whole simulation thus gives a much larger sample size for the performance data of the same system configuration setting. There is no necessity to further distinguish the system configuration for the server in the performance data analysis.

The analysis approach on the client performance evaluation is slightly different from the server and is lengthier. Heterogeneous client system configuration is the main reason for the slight difference. We will not include the finding we have on the performance of remote client in this paper because of the limited page count. Please refer to [2] for detailed treatment and analysis. In this paper, we solely concentrate on the performance of digital cash server.

5. Results and Analysis

The computation time has direct relationship with the bit length. The bit length, in the context of this paper, implicitly refers to the bit length of parameter p, unless otherwise stated. Table 3 gives the computation performance of the server collected from the simulation. The control variable in the Table 3 is parameter length located at the left hand side. The right hand side portion of the table lists the average computation time, in millisecond, taken by the server to complete each protocol.

The protocols share a similar pattern in the computation time distribution. Basically there are three different phases in all protocol computations: 512-1024, 1024-2048 and 2048-4096 bits. Each phase has steeper increasing rate, compared to the previous phase. Preliminary study indicates that the computation time could be approximately modelled using second or third degree polynomial function.

The setup computation consumes the most computation time. The setup computation time for the bit length of 1024 bits and below is less than 2 seconds (or 2000 milliseconds) on average. It is relatively smaller and unnoticeable by the user. Therefore, for frequent and small amount of digital cash usage, it is desirable to use the bit length of 1024 bits.

Although the other protocols (Withdrawal, Payment and Deposit) share the same changing pattern with the Setup protocol in the computation distribution, their computation times are much smaller compared to setup protocol, especially for the range between 1024 and 4096 bits.

To achieve a balance point between security and efficiency, 2048-bit is a good choice. For the case of 2048 bits, the average computation times for all protocols (except setup protocol) are less than 1 second. Although the setup protocol requires an average computation time of 25035.433 ms (around 25 seconds), it is still practical to settle down with 2048-bit, as setup protocol needs to be done only once for each new user. The communication does not change as much as the computation does in setup protocol. From the Table 4, the communication time for setup and withdrawal protocols constantly increases (although not obvious) with the growth of bit lengths. On the contrary, the communication time for payment and deposit protocols has irregular pattern that does not absolutely follows the growth of bit length. Specifically, the deposit communication time has huge standard deviation indicating its distribution is wide.

The parameter length is fixed at 1024 bits for the following analysis on thread priority. There is a total of 4199 records fallen into this category for the server. Moving from top to bottom of the Table 5, the overall priority level is increasing. The hypothesis states that the computation performance increases (smaller computation time) with the higher priority.

However, the analysis result shows that higher thread priority does not have significant influence to the computation time. The transition to higher priority does not impose noticeable performance improvement. On the other hand, the lowest thread priority obviously drags down the computation performance for all protocols

Analysis on Table 6 further indicates that the standard deviation for communication time is unpredictable (with no observable patterns) and is relatively higher than the mean value. This phenomenon implies that the distribution of the communication time samples is very wide. Basically, the average communication time is becoming lower although not obvious across the priority transition.

6. Conclusion

The analysis of this paper shows that the Brands' digital cash system is feasible to work in TCP/IP networking environment from the viewpoint of digital cash server. The transaction of digital cash system does not consume obvious or noticeable long time to complete in terms of computation and communication. This could give the users a comfortable experience to use digital cash as their electronic payment method, which provides high-degree security, convenience and anonymity.

References

[1] S. Brands, "An Efficient Offline Electronic Cash System Based On The Representation Problem." CWI Technical Report CS-R9323, 1993.

[2] P. C. Teo, "Implementation and Performance Evaluation of Brands' Digital Cash System in TCP/IP Environment." Master's Thesis, Universiti Teknologi Malaysia, 2002.

[3] P. C. Teo, S. Ahmad Zuri, M. Y. Zulkalnain, "Implementation of Brands' Digital Cash System in TCP/IP Networking Environment." M2USIC 2002, MMU International Symposium on Information and Communications Technologies, 2002.

[4] C. P. Schnorr, "Efficient Identification and Signatures for Smart Cards." Advances in Cryptology, Crypto'89. Springer-Verlag, 1990.

[5] National Institute of Standards and Technology, "Digital Signature Standard" U.S.: (NIST FIPS PUB 186), 1994.

Table 3. Server's Average Computation Time (ms) for Different Parameter Length

Parameter Length (bit)	Setup	Withdrawal	Payment	Deposit
512	207.95	10.71	20.12	91.44
768	1421.29	50.05	95.96	171.08
1024	1921.07	67.48	127.22	222.21
1536	15507.60	303.72	556.45	642.71
2048	25035.44	439.95	812.02	889.71
3072	138179.49	1864.69	3174.45	3279.05
4096	288299.19	2625.15	4490.91	4572.32

Sample Size: 6248

Table 4. Server's Average Communication Time (ms) for Different Parameter Lengths

Parameter Length (bit)	Setup	Withdrawal	Payment	Deposit
512	1380.23	1373.20	1704.81	803.27
768	1768.36	1812.12	1717.13	1186.39
1024	1710.24	1926.74	1534.38	922.69
1536	2564.23	2413.74	1943.84	1106.19
2048	2710.15	3123.41	2117.48	1279.05
3072	3900.23	5749.31	1649.64	1681.29
4096	5316.58	6689.30	1696.88	1064.60

Sample Size: 6248

Table 5. Server's Average Computation/Communication Time (ms) for Different Thread Priority

Thread Priority	Setup		Withdrawal		Payment		Deposit	
0	Comput-ation	Commun-ication	Comput-tation	Commun-ication	Comput-ation	Commun-ication	Compu-tation	Commun-nication
1	3612.45	1882.319	89.00	2371.632	183.43	1494.47	407.22	1224.47
2	1697.90	1964.222	63.92	2780.556	127.85	1557.95	200.70	1116.95
3	1815.59	2412.806	65.06	3043.910	122.98	1779.29	197.41	1590.66
4	1978.71	2484.377	93.25	2762.192	218.11	2129.89	599.40	1243.78
5	1957.34	1954.400	69.81	2284.025	128.90	1785.53	228.87	1150.11
Sub Average	1858.06	2060.833	67.32	1981.750	129.77	1662.37	210.97	947.85
0	2166.18	2136.470	75.24	2562.846	153.72	1738.92	315.02	1222.10
1	3437.54	2068.046	92.46	2532.192	175.62	1557.48	383.93	1239.41
2	1784.86	2254.577	65.17	2438.300	124.27	2084.90	197.73	1209.90
3	1743.84	2146.215	63.25	2490.346	127.10	1923.00	209.67	1065.13
4	1824.40	1545.137	65.77	1661.316	117.72	1426.10	190.49	746.62
5	1947.80	1444.208	64.40	1509.438	122.52	1378.21	195.41	751.23
6	1700.80	1329.608	63.92	1793.300	121.00	1361.96	191.44	833.20
Sub Average	1826.90	1358.283	60.87	1647.709	117.33	1388.09	182.63	804.60
1	1895.50	1608.589	66.51	1777.512	121.59	1480.21	200.86	814.26
2	1732.46	1913.192	63.40	1785.275	121.99	1626.30	193.74	980.09
3	1713.05	1588.485	61.70	1904.346	117.00	1570.40	189.93	1536.54
4	1653.71	1563.885	61.63	1847.215	115.51	1510.46	200.46	791.99
5	1588.06	1601.098	60.75	1548.159	116.00	1344.50	203.00	802.31
Sub Average	1894.86	1341.385	60.96	1655.254	116.08	1474.57	183.46	893.831
Grand Average	1715.78	1596.754	61.66	1746.847	117.24	1502.86	194.15	1000.662
	1921.07	1710.243	67.48	1926.749	127.22	1534.38	222.21	922.696

Sample Size: 4199; Note: The parameter length is 1024 bits